Robust Machine Learning: Detection, Evaluation and Adaptation Under Distribution Shift

Saurabh Garg

May 2024
CMU-ML-24-106

Machine Learning Department
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Zachary C. Lipton (CMU), Co-chair
Sivaraman Balakrishnan (CMU), Co-chair
Zico Kolter (CMU)
Aditi Raghunathan (CMU)
Ludwig Schmidt (Anthropic, Stanford)

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

This research was partially supported by graduate fellowships from Amazon, JP Morgan and Bloomberg.

The views and conclusions contained in this document are those of the author only and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. government, or any other entity.

Keywords: Distribution shift, statistic machine learning, label shift, covariate shift, model evaluation, domain adaptation, online learning, vision language models, spurious correlations, calibration

In memory of my grandparents,
who have shaped me in countless unseen ways throughout my life...

Abstract

Deep learning, despite its broad applicability, grapples with robustness challenges in real-world applications, especially when training and test distributions differ. Reasons for the discrepancy between training and test distributions include gradual changes in human behavior or differences in the demographics of the environment where the service is being used. While obtaining labeled data for anticipated distribution shifts can be daunting, unlabeled samples are relatively cheap and abundantly available.

My research leverages unlabeled data from the target domain to identify structural relationships between the target and source domains, and then use them to adapt and evaluate models. The work discussed in thesis involves understanding the behavior of deep models, both theoretically and empirically, and using those insights to develop robust methods. In particular, this thesis surveys my work on the following three questions:

Q1: How to adapt models in the face of distribution shifts? Absent assumptions on the nature of the distribution shifts, this task is impossible. My research in this direction is focused on formulating assumptions on distribution shift scenarios appearing in the wild and developing procedures that improve and adapt deep models under those shifts by leveraging unlabeled data. Part I and II of this thesis delve into this research.

Q2: How can we evaluate models' performance without access to labeled data? Deep learning models fail silently, i.e., they cannot flag uncertain decisions. To build reliable machine learning systems, obtaining certificates for accuracy is as important as robustifying these systems. Part III discusses my research in this direction and presents techniques that leverage unlabeled data to predict model accuracy.

Q3: How can we leverage foundation models to address distribution shift challenges? Foundation models, such as vision language models, have demonstrated impressive performance on a wide range of tasks. However, these models also lack robustness due to spurious correlations, poor image-text alignment, etc. Moreover, they also get outdated as the internet data evolves presenting novel challenges in keeping them up-to-date. Part IV of my thesis discusses my work on understanding the behavior of foundation models and developing techniques to improve their robustness under distribution shifts.

Overall, this thesis expands the frontier of robust machine learning by developing techniques that leverage unlabeled data to adapt and evaluate models under distribution shifts. The work presented here is a step towards developing a comprehensive toolkit for robust machine learning in the face of distribution shifts.

Contents

Chapter 1

Introduction

Deep learning has seen remarkable success in various applications, yet it faces significant challenges when deployed in real-world settings where training and test distributions diverge. This discrepancy can arise due to evolving human behaviors, demographic shifts, or changing environmental factors where the model is used. Adapting deep learning models to these distribution shifts is crucial for maintaining performance and reliability in deployment.

In response to these challenges, leveraging unlabeled data from the target domain has emerged as a promising strategy. Unlabeled data is often more accessible and cost-effective compared to obtaining labeled samples for every distribution shift. This approach forms the foundation of research aimed at improving the robustness and adaptability of deep learning models under changing conditions. My research focuses on harnessing unlabeled data to understand and address distribution shift challenges in deep learning. The thesis explores methodologies to identify structural relationships between source and target domains using unlabeled data, subsequently using this knowledge to adapt and evaluate models.

1.1 Organization

The primary objective of this thesis is to develop techniques that enhance the robustness of machine learning models against distribution shifts. The work presented here is structured as follows:

Q1: *How to adapt models in response to distribution shifts?* This research investigates assumptions and procedures necessary for improving and adapting deep learning models under distribution shifts, leveraging insights from unlabeled data. Part I discusses adaptation under label shift scenarios and Part II discusses distribution shift scenarios involving input distribution shifts and relaxed label shift scenarios.

- Chapter 2 presents a unified view on the problem of label shift estimation.
- Chapter 3 introduces an online label shift adaptation problem and presents algorithms for this setting.

- Chapter 4 and Chapter 5 extends the work on label shift to scenarios where along with shifts in label marginal, previously unseen classes may appear.
- Chapter 6 discusses the benefits of combining contrastive learning and self-training under distribution shift.
- Chapter 7 introduces a benchmark for domain adaptation under relaxed label shift highlighting the brittle nature of existing methods in presence of label proportion shifts and extends our label shift methods to this setting.

Q2: *How can we evaluate models' performance without access to labeled data?* Deep learning models often lack mechanisms to detect uncertain decisions, leading to silent failures under distribution shifts. Part III of this thesis explores techniques to predict model accuracy using unlabeled data, crucial for building reliable machine learning systems.

- Chapter 8 discusses the problem of evaluating models without access to labeled data and proposes a method to guarantee in-distribution generalization.
- Chapter 9 and Chapter 10 discusses the problem of predicting out-of-distribution performance without access to labeled data.

Q3: *How can we leverage foundation models to address distribution shift challenges?* Foundation models, though powerful, exhibit vulnerabilities under distribution shifts due to spurious correlations or outdated data. Part IV of this thesis investigates strategies to enhance the robustness of foundation models in evolving environments.

- Chapter 11 highlights temporal distribution shift problems with OpenAI CLIP models and propose a continual learning benchmark with 12.7 B image-text pairs with time metadata for continual training of CLIP.
- Chapter 12 discusses the brittle nature foundation models to tasks with spurious correlations and proposes Prompting for Robustness (PfR) to leverage language descriptions of spurious attributes to train robust foundation models.

The work presented in this thesis contributes to expanding the frontier of robust machine learning by developing novel techniques that harness the potential of unlabeled data to adapt and evaluate models under distribution shifts. This research aims to provide a comprehensive toolkit for deploying reliable machine learning systems in dynamic real-world settings. By addressing distribution shift challenges, this thesis aims to bridge the gap between theoretical insights and practical methodologies, advancing the field of robust machine learning.

Part I

Adaptation Under Label Shift

Chapter 2

A Unified View of Label Shift Estimation

Based on Garg et al. (2020a): Saurabh Garg, Yifan Wu, Sivaraman Balakrishnan, and Zachary Lipton. A Unified View of Label Shift Estimation. Advances in Neural Information Processing Systems (NeurIPS), 2020.

Abstract

Under label shift, the label distribution $p(y)$ might change but the class-conditional distributions $p(x|y)$ do not. There are two dominant approaches for estimating the label marginal. BBSE, a moment-matching approach based on confusion matrices, is provably consistent and provides interpretable error bounds. However, a maximum likelihood estimation approach, which we call MLLS, dominates empirically. In this chapter, we present a unified view of the two methods and the first theoretical characterization of MLLS. Our contributions include (i) consistency conditions for MLLS, which include calibration of the classifier and a confusion matrix invertibility condition that BBSE also requires; (ii) a unified framework, casting BBSE as roughly equivalent to MLLS for a particular choice of calibration method; and (iii) a decomposition of MLLS's finite-sample error into terms reflecting miscalibration and estimation error. Our analysis attributes BBSE's statistical inefficiency to a loss of information due to coarse calibration. Experiments on synthetic data, MNIST, and CIFAR10 support our findings.

2.1 Introduction

This chapter focuses on *label shift* (Lipton et al., 2018b; Saerens et al., 2002; Storkey, 2009), which aligns with the *anticausal* setting in which the labels y cause the features x (Schölkopf et al., 2012). Label shift arises in diagnostic problems because diseases cause symptoms. In this setting, an intervention on $p(y)$ induces the shift, but the process generating x given

y is fixed ($p_s(x|y) = p_t(x|y)$). Under label shift, the optimal predictor may change, e.g., the probability that a patient suffers from a disease given their symptoms can increase under a pandemic. Contrast label shift with the better-known *covariate shift* assumption, which aligns with the assumption that x causes y, yielding the reverse implication that $p_s(y|x) = p_t(y|x)$.

Under label shift, our first task is to estimate the ratios $w(y) = p_t(y)/p_s(y)$ for all labels y. Two dominant approaches leverage off-the-shelf classifiers to estimate w: (i) *Black Box Shift Estimation* (BBSE) (Lipton et al., 2018b) and a variant called *Regularized Learning under Label Shift* (RLLS) (Azizzadenesheli et al., 2019): moment-matching based estimators that leverage (possibly biased, uncalibrated, or inaccurate) predictions to estimate the shift; and (ii) Maximum Likelihood Label Shift (MLLS) (Saerens et al., 2002): an Expectation Maximization (EM) algorithm that assumes access to a classifier that outputs the true source distribution conditional probabilities $p_s(y|x)$.

Given a predictor $\widehat{f}$ with an invertible confusion matrix, BBSE and RLLS have known consistency results and finite-sample guarantees (Azizzadenesheli et al., 2019; Lipton et al., 2018b). However, MLLS, in combination with a calibration heuristic called Bias-Corrected Temperature Scaling (BCTS), outperforms them empirically (Alexandari et al., 2021).

In this chapter, we theoretically characterize MLLS, establishing conditions for consistency and bounding its finite-sample error. To start, we observe that given the true label conditional $p_s(y|x)$, MLLS is simply a concave Maximum Likelihood Estimation (MLE) problem and standard results apply. However, because we never know $p_s(y|x)$ exactly, MLLS is always applied with an estimated model $\widehat{f}$ and thus the procedure consists of MLE under model misspecification.

First, we prove that (i) *canonical calibration* (Definition 2.2.1) and (ii) an invertible confusion matrix (as required by BBSE) are *sufficient conditions* to ensure MLLS's consistency (Proposition 2.4.4, Theorems H.2.1 and 2.4.3). We also show that calibration can sometimes be *necessary* for consistency (Example 1 in Section 2.4.3). Recall that neural network classifiers tend to be uncalibrated absent post-hoc adjustments (Guo et al., 2017). Second, we observe that confusion matrices can be instruments for calibrating a classifier. Applying MLLS with this technique, BBSE and MLLS are distinguished only by their objective functions. Through extensive experiments, we show that they perform similarly, concluding that MLLS's superior performance (when applied with more granular calibration techniques) is not due to its objective but rather to the information lost by BBSE via confusion matrix calibration. Third, we analyze the finite-sample error of the MLLS estimator by decomposing its error into terms reflecting the miscalibration error and finite-sample error (Theorem 2.5.4). Depending on the calibration method, the miscalibration error can further be divided into two terms: finite sample error due to re-calibration on a validation set and the minimum achievable calibration error with that technique.

We validate our results on synthetic data, MNIST, and CIFAR-10. Empirical results show that MLLS can have 2–10× lower Mean Squared estimation Error (MSE) depending on the magnitude of the shift. Our experiments relate MLLS's MSE to the granularity of the calibration.

In summary, we contribute the following: (i) Sufficient conditions for MLLS's consistency; (ii) Unification of MLLS and BBSE methods under a common framework, with BBSE corresponding to a particular choice of calibration method; (iii) Finite-sample error bounds for MLLS; (iv) Experiments on synthetic and image recognition datasets that support our theoretical arguments.

2.2 Problem Setup

Let $\mathcal{X}$ be the input space and $\mathcal{Y} = \{1, 2, \ldots, k\}$ the output space. Let $\mathrm{P}_s, \mathrm{P}_t : \mathcal{X} \times \mathcal{Y} \to [0, 1]$ be the source and target distributions and let p_s and p_t denote the corresponding probability density (or mass) functions. We use $\mathbb{E}_s$ and $\mathbb{E}_t$ to denote expectations over the source and target distributions. In unsupervised domain adaptation, we possess labeled source data $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ and unlabeled target data $\{x_{n+1}, x_{n+2}, \ldots, x_{n+m}\}$. We also assume access to a black-box predictor $\widehat{f} : \mathcal{X} \mapsto \Delta^{k-1}$, e.g., a model trained to approximate the true probability function f^*, where $f^*(x) := p_s(\cdot|x)$. Here and in the rest of the paper, we use Δ^{k-1} to denote the standard k-dimensional probability simplex. For a vector v, we use v_y to access the element at index y.

Absent assumptions relating the source and target distributions, domain adaptation is underspecified (Ben-David et al., 2010c). We work with the *label shift* assumption, i.e., $p_s(x|y) = p_t(x|y)$, focusing on multiclass classification. Moreover, we assume non-zero support for all labels in the source distribution: for all $y \in \mathcal{Y}$, $p_s(y) \geqslant c > 0$ (Azizzadenesheli et al., 2019; Lipton et al., 2018b). Under label shift, three common goals are (i) detection—determining whether distribution shift has occurred; (ii) quantification—estimating the target label distribution; and (iii) correction—producing a predictor that minimizes error on the target distribution (Lipton et al., 2018b).

This paper focuses on goal (ii), estimating importance weights $w(y) = p_t(y)/p_s(y)$ for all $y \in \mathcal{Y}$. Given w, we can update our classifiers on the fly, either by retraining in an importance-weighted ERM framework (Azizzadenesheli et al., 2019; Gretton et al., 2009; Lipton et al., 2018b; Shimodaira, 2000)—a practice that may be problematic for overparameterized neural networks (Byrd and Lipton, 2019), or by applying an analytic correction (Alexandari et al., 2021; Saerens et al., 2002). Within the ERM framework, the generalization result from Azizzadenesheli et al. (2019) (Theorem 1) depends only on the error of the estimated weights, and hence any method that improves weight estimates tightens this bound.

There are multiple definitions of calibration in the multiclass setting. Guo et al. (2017) study the calibration of the arg-max prediction, while Kumar et al. (2019) study a notion of per-label calibration. We use canonical calibration (Vaicenavicius et al., 2019) and the expected canonical calibration error on the source data defined as follows:

Definition 2.2.1 (Canonical calibration)**.** *A prediction model $f : \mathcal{X} \mapsto \Delta^{k-1}$ is canonically calibrated on the source domain if for all $x \in \mathcal{X}$ and $j \in \mathcal{Y}$, $\mathrm{P}_s(y = j | f(x)) = f_j(x)$.*

Definition 2.2.2 (Expected canonical calibration error)**.** *For a predictor f, the expected squared canonical calibration error on the source domain is $\mathcal{E}^2(f) = \mathbb{E}_s \left\|f - f_c\right\|^2$, where $f_c = \mathrm{P}_s(y = \cdot|f(x))$.*

Calibration methods typically work either by calibrating the model during training or by calibrating a trained classifier on held-out data, post-hoc. We refer the interested reader to Kumar et al. (2019) and Guo et al. (2017) for detailed studies on calibration. We focus on the latter category of methods. Our experiments follow Alexandari et al. (2021), who leverage BCTS [1] to calibrate their models. BCTS extends temperature scaling (Guo et al., 2017) by incorporating per-class bias terms.

2.3 Prior Work

Two families of solutions have been explored that leverage a blackbox predictor: BBSE (Lipton et al., 2018b), a moment matching method, uses the predictor $\widehat{f}$ to compute a confusion matrix $C_{\widehat{f}} := p_s(\widehat{y}, y) \in \mathbb{R}^{k\times k}$ on the source data. Depending on how $\widehat{y}$ is defined, there are two types of confusion matrix for a predictor $\widehat{f}$: (i) the *hard confusion matrix* $\widehat{y} = \arg\max \widehat{f}(x)$; and (ii) the *soft confusion matrix*, where $\widehat{y}$ is defined as a random prediction that follows the discrete distribution $\widehat{f}(x)$ over $\mathcal{Y}$. Both soft and hard confusion matrix can be estimated from labeled source data samples. The estimate $\widehat{w}$ is computed as $\widehat{w} := \widehat{C}_{\widehat{f}}^{-1}\widehat{\mu}$, where $\widehat{C}_{\widehat{f}}$ is the estimate of confusion matrix and $\widehat{\mu}$ is an estimate of $p_t(\widehat{y})$, computed by applying the predictor $\widehat{f}$ to the target data. In a related vein, RLLS (Azizzadenesheli et al., 2019) incorporates an additional regularization term of the form $\|w - 1\|$ and solves a constrained optimization problem to estimate the shift ratios w.

MLLS estimates w as if performing maximum likelihood estimation, but substitutes the predictor outputs for the true probabilities $p_s(y|x)$. Saerens et al. (2002), who introduce this procedure, describe it as an application of EM. However, as observed in (Alexandari et al., 2021; Du Plessis and Sugiyama, 2014b), the likelihood objective is concave, and thus a variety of optimization algorithms may be applied to recover the MLLS estimate. Alexandari et al. (2021) also showed that MLLS underperforms BBSE when applied naively, a phenomenon that we shed more light on in this paper.

2.4 A Unified View of Label Shift Estimation with Black Box Predictors

We now present a unified view that subsumes MLLS and BBSE and demonstrate how each is instantiated under this framework. We also establish identifiability and consistency conditions for MLLS, deferring a treatment of finite-sample issues to Section 2.5. For convenience, throughout Sections 3 and 4, we use the term *calibration* exclusively to refer

[1] Motivated by the strong empirical results in Alexandari et al. (2021), we use BCTS in our experiments as a surrogate to canonical calibration.

to canonical calibration (Definition 2.2.1) on the source data. We relegate all technical proofs to Appendix A.4.

2.4.1 A Unified Distribution Matching View

To start, we introduce a *generalized* distribution matching approach for estimating w. Under label shift, for any (possibly randomized) mapping from $\mathcal{X}$ to $\mathcal{Z}$, we have that $p_s(z|y) = p_t(z|y)$ since, $p_s(z|y) = p_t(z|y) = \int_{\mathcal{X}} p(z|x)p(x|y)dx$. Throughout the paper, we use the notation $p(z|y)$ to represent either $p_s(z|y)$ or $p_t(z|y)$ (which are identical). We now define a family of distributions over $\mathcal{Z}$ parameterized by $w \in \mathcal{W}$ as

$$p_w(z) = \sum\nolimits_{y=1}^{k} p(z|y)p_s(y)w_y = \sum\nolimits_{y=1}^{k} p_s(z,y)w_y, \tag{2.1}$$

where $\mathcal{W} = \{w \mid \forall y\,, w_y \geqslant 0 \text{ and } \sum_{y=1}^{k} w_y p_s(y) = 1\}$. When $w = w^*$, we have that $p_w(z) = p_t(z)$. For fixed $p(z|x)$, $p_t(z)$ and $p_s(z,y)$ are known because $p_t(x)$ and $p_s(x,y)$ are known. So one potential strategy to estimate w^* is to find a weight vector w such that

$$\sum\nolimits_{y=1}^{k} p_s(z,y)w_y = p_t(z) \qquad \forall z \in \mathcal{Z}\,. \tag{2.2}$$

At least one such weight vector w must exist as w^* satisfies (2.2). We now characterize conditions under which the weight vector w satisfying (2.2) is unique:

Lemma 2.4.1 (Identifiability)**.** *If the set of distributions $\{p(z|y) : y = 1, ..., k\}$ are linearly independent, then for any w that satisfies (2.2), we must have $w = w^*$. This condition is also necessary in general: if the linear independence does not hold then there exists a problem instance where we have $w, w^* \in \mathcal{W}$ satisfying (2.2) while $w \neq w^*$.*

Lemma 2.4.1 follows from the fact that (2.2) is a linear system with at least one solution w^*. This solution is unique when $p_s(z,y)$ is of rank k. The linear independence condition in Lemma 2.4.1, in general, is sufficient for identifiability of discrete $\mathcal{Z}$. However, for continuous $\mathcal{Z}$, the linear dependence condition has the undesirable property of being sensitive to changes on sets of measure zero. By changing a collection of linearly dependent distributions on a set of measure zero, we can make them linearly independent. As a consequence, we impose a *stronger* notion of identifiability i.e., the set of distributions $\{p(z|y) : y = 1, ..., k\}$ are such that there does not exist $v \neq 0$ for which $\int_{\mathcal{Z}} |\sum_y p(z|y)v_y| dz = 0$. We refer this condition as *strict linear independence*.

In generalized distribution matching, one can set $p(z|x)$ to be the Dirac delta function at δ_x[2] such that $\mathcal{Z}$ is the same space as $\mathcal{X}$, which leads to solving (2.2) with z replaced by x. In practice where $\mathcal{X}$ is high-dimensional and/or continuous, approximating the solution to (2.2) from finite samples can be hard when choosing $z = x$. Our motivation for generalizing distribution matching from $\mathcal{X}$ to $\mathcal{Z}$ is that the solution to (2.2) can be better approximated using finite samples when $\mathcal{Z}$ is chosen carefully. Under this framework, the design of a label shift estimation algorithm can be decomposed into two parts: (i) the choice of $p(z|x)$ and (ii) how to approximate the solution to (2.2). Later on, we consider how these design choices may affect label shift estimation procedures in practice.

[2] For simplicity we will use $z = x$ to denote that $p(z|x) = \delta_x$.

2.4.2 The Confusion Matrix Approach

If $\mathcal{Z}$ is a discrete space, one can first estimate $p_s(z, y) \in \mathbb{R}^{|\mathcal{Z}| \times k}$ and $p_t(z) \in \mathbb{R}$, and then subsequently attempt to solve (2.2). Confusion matrix approaches use $\mathcal{Z} = \mathcal{Y}$, and construct $p(z|x)$ using a black box predictor $\widehat{f}$. There are two common choices to construct the confusion matrix: (i) The soft confusion matrix approach: We set $p(z|x) := \widehat{f}(x) \in \Delta^{k-1}$. We then define a random variable $\widehat{y} \sim \widehat{f}(x)$ for each x. Then we construct $p_s(z, y) = p_s(\widehat{y}, y)$ and $p_t(z) = p_t(\widehat{y})$. (ii) The hard confusion matrix approach: Here we set $p(z|x) = \delta_{\arg\max \widehat{f}(x)}$. We then define a random variable $\widehat{y} = \arg\max \widehat{f}(x)$ for each x. Then again we have $p_s(z, y) = p_s(\widehat{y}, y)$ and $p_t(z) = p_t(\widehat{y})$.

Since $p_s(z, y)$ is a square matrix, the identifiability condition becomes the invertibility of the confusion matrix. Given an estimated confusion matrix, one can find w by inverting the confusion matrix (BBSE) or minimizing some distance between the vectors on the two sides of (2.2).

2.4.3 Maximum Likelihood Label Shift Estimation

When $\mathcal{Z}$ is a continuous space, the set of equations in (2.2) indexed by $\mathcal{Z}$ is intractable. In this case, one possibility is to find a weight vector $\widetilde{w}$ by minimizing the KL-divergence $\mathrm{KL}(p_t(z), p_w(z)) = \mathbb{E}_t \left[\log p_t(z)/p_w(z)\right]$, for p_w defined in (2.1). This is equivalent to maximizing the population log-likelihood: $\widetilde{w} := \arg\max_{w \in \mathcal{W}} \mathbb{E}_t \left[\log p_w(z)\right]$. One can further show that $\mathbb{E}_t \left[\log p_w(z)\right] = \mathbb{E}_t[\log \sum_{y=1}^k p_s(z, y) w_y] = \mathbb{E}_t[\log \sum_{y=1}^k p_s(y|z) p_s(z) w_y] = \mathbb{E}_t[\log \sum_{y=1}^k p_s(y|z) w_y] + \mathbb{E}_t \left[\log p_s(z)\right]$. Therefore we can equivalently define:

$$\widetilde{w} := \arg\max_{w \in \mathcal{W}} \mathbb{E}_t \Big[\log \sum\nolimits_{y=1}^k p_s(y|z) w_y \Big] . \tag{2.3}$$

This yields a straightforward convex optimization problem whose objective is bounded from below (Alexandari et al., 2021; Du Plessis and Sugiyama, 2014b). Assuming access to labeled source data and unlabeled target data, one can maximize the empirical counterpart of the objective in (2.3), using either EM or an alternative iterative optimization scheme. Saerens et al. (2002) derived an EM algorithm to maximize the objective (2.3) when $z = x$, assuming access to $p_s(y|x)$. Absent knowledge of the ground truth $p_s(y|x)$, we can plug in any approximate predictor f and optimize the following objective:

$$w_f := \arg\max_{w \in \mathcal{W}} \mathcal{L}(w, f) := \arg\max_{w \in \mathcal{W}} \mathbb{E}_t \left[\log f(x)^T w \right] . \tag{2.4}$$

In practice, f is fit from a finite sample drawn from $p_s(x, y)$ and standard machine learning methods often produce uncalibrated predictors. While BBSE and RLLS are provably consistent whenever the predictor f yields an invertible confusion matrix, to our knowledge, no prior works have established sufficient conditions to guarantee MLLS' consistency when f differs from $p_s(y|x)$.

It is intuitive that for some values of $f \neq p_s(y|x)$, MLLS will yield inconsistent estimates. Supplying empirical evidence, Alexandari et al. (2021) show that MLLS performs poorly when f is a vanilla neural network predictor learned from data. However, Alexandari et al. (2021) also show that in combination with a particular post-hoc calibration technique, MLLS achieves low error, significantly outperforming BBSE and RLLS. As the calibration error is not a distance metric between f and $p_s(y|x)$ (zero calibration error does not indicate $f = p_s(y|x)$), a calibrated predictor f may still be substantially different from $p_s(y|x)$. Some natural questions then arise:

1. *Why does calibration improve MLLS so dramatically?*
2. *Is calibration necessary or sufficient to ensure the consistency of MLLS?*
3. *What accounts for the comparative efficiency of MLLS over BBSE?* (Addressed in Section 2.5)

To address the first two questions, we make the following observations. Suppose we define z (for each x) with distribution $p(z|x) := \delta_{f(x)}$, for some calibrated predictor f. Then, because f is calibrated, it holds that $p_s(y|z) = f(x)$. Note that in general, the MLLS objective (2.4) can differ from (2.3). However, when $p(z|x) := \delta_{f(x)}$, the two objectives are identical. We can formalize this as follows: If f is calibrated, then the two objectives (2.3) and (2.4) are identical when $\mathcal{Z}$ is chosen as Δ^{k-1} and $p(z|x)$ is defined to be $\delta_{f(x)}$. Lemma 2.4.3 follows from changing the variable of expectation in (2.4) from x to $f(x)$ and applying $f(x) = p_s(y|f(x))$ (definition of calibration). It shows that MLLS with a calibrated predictor on the input space $\mathcal{X}$ is in fact equivalent to performing distribution matching in the space $\mathcal{Z}$. Building on this observation, we now state our population-level consistency theorem for MLLS:

Theorem 2.4.2 (Population consistency of MLLS)**.** *If a predictor $f : \mathcal{X} \mapsto \Delta^{k-1}$ is calibrated and the distributions $\{p(f(x)|y) : y = 1, \ldots, k\}$ are strictly linearly independent, then w^* is the unique maximizer of the MLLS objective* (2.4).

We now turn our attention to establishing consistency of the sample-based estimator. Let $x_1, x_2, \ldots, x_m \overset{iid}{\sim} p_t(x)$. The finite sample objective for MLLS can be written as

$$\widehat{w}_f := \underset{w\in\mathcal{W}}{\arg\max}\, \frac{1}{m}\sum\nolimits_{i=1}^{m} \log f(x_i)^T w := \underset{w\in\mathcal{W}}{\arg\max}\, \mathcal{L}_m(w, f)\,. \tag{2.5}$$

Theorem 2.4.3 (Consistency of MLLS)**.** *If f satisfies the conditions in Theorem H.2.1, then $\widehat{w}_f$ in* (2.5) *converges to w^* almost surely.*

The main idea of the proof of Theorem 2.4.3 is to derive a metric entropy bound on the class of functions $\mathcal{G} = \left\{(f^T w)/(f^T w + f^T w^*)|w \in \mathcal{W}\right\}$ to prove Hellinger consistency (Theorem 4.6 (van de Geer, 2000)). The consistency of MLLS relies on the linear independence of the collection of distributions $\{p(f(x)|y) : y = 1, \ldots, k\}$. The following result develops several alternative equivalent characterizations of this linear independence condition.

Proposition 2.4.4. *For a calibrated predictor f, the following statements are equivalent:*

(1) $\{p(f(x)|y) : y = 1, \ldots, k\}$ *are strictly linearly independent.*

(2) $\mathbb{E}_s\left[f(x)f(x)^T\right]$ *is invertible.*

(3) The soft confusion matrix of f is invertible.

Proposition 2.4.4 shows that with a calibrated predictor, the invertibility condition as required by BBSE (or RLLS) is exactly the same as the linear independence condition required for MLLS's consistency.

Having provided sufficient conditions, we consider a binary classification example to provide intuition for why we need calibration for consistency. In this example, we relate the estimation error to the miscalibration error, showing that calibration is not only sufficient but also necessary to achieve zero estimation error for a certain class of predictors.

Example 1. Consider a mixture of two Gaussians with $p_s(x|y = 0) := \mathcal{N}(\mu, 1)$ and $p_s(x|y = 1) := \mathcal{N}(-\mu, 1)$. We suppose that the source mixing coefficients are both $\frac{1}{2}$, while the target mixing coefficients are $\alpha(\neq \frac{1}{2}), 1 - \alpha$. Assume a class of probabilistic threshold classifiers: $f(x) = [1 - c, c]$ for $x \geqslant 0$, otherwise $f(x) = [c, 1 - c]$ with $c \in [0, 1]$. Then the population error of MLLS is given by

$$4\left|\frac{(1-2\alpha)(p_s(x \geqslant 0|y=0) - c)}{1-2c}\right|,$$

which is zero only if $c = p_s(x \geqslant 0|y = 0)$ for a non-degenerate classifier.

The expression for estimation error arising from our example yields two key insights: (i) an uncalibrated thresholded classifier has an estimation error proportional to the true shift in label distribution i.e. $1 - 2\alpha$; (ii) the error is also proportional to the canonical calibration error which is $p_s(x \geqslant 0|y = 0) - c$. While earlier in this section, we concluded that canonical calibration is sufficient for consistency, the above example provides some intuition for why it might also be necessary. In Appendix A.3, we show that marginal calibration (Guo et al., 2017; Kumar et al., 2019; Vaicenavicius et al., 2019), a less restricted definition is insufficient to achieve consistency.

2.4.4 MLLS with Confusion Matrix

So far, we have shown that MLLS with any calibrated predictor can be viewed as distribution matching in a latent space. Now we discuss a method to construct a predictor f to perform MLLS given any $p(z|x)$, e.g., those induced by confusion matrix approaches. Recall, we already have the maximum log-likelihood objective. It just remains to construct a calibrated predictor f from the confusion matrix.

This is straightforward when $p(z|x)$ is deterministic, i.e., $p(z|x) = \delta_{g(x)}$ for some function g: setting $f(x) = p_s(y|g(x))$ makes the objectives (2.3) and (2.4) to be the same. Recall that for the hard confusion matrix, the induced latent space is $p(z|x) = \delta_{\arg\max \hat{f}(x)}$. So the corresponding predictor in MLLS is $f(x) = p_s(y|\widehat{y}_x)$, where $\widehat{y}_x = \arg\max \hat{f}(x)$. Then we obtain the MLLS objective for the hard confusion matrix:

$$\max_{w \in \mathcal{W}} \mathbb{E}_t \left[\log \sum\nolimits_{y=1}^{k} p_s(y|\widehat{y}_x) w_y\right] . \tag{2.6}$$

The confusion matrix $C_{\widehat{f}}$ and predictor $\widehat{f}$ directly give us $p_s(y|\widehat{y}_x)$. Given an input x, one can first get $\widehat{y}_x$ from $\widehat{f}$, then normalize the $\widehat{y}_x$-th row of $C_{\widehat{f}}$ as $p_s(y|\widehat{y}_x)$. We denote MLLS with hard confusion matrix calibration (2.6) by MLLS-CM.

When $p_s(z|x)$ is stochastic, we need to extend (2.4) to allow f to be a random predictor: $f(x) = p_s(y|z)$ for $z \sim p(z|x)$[3]. To incorporate the randomness of f, one only needs to change the expectation in (2.4) to be over both x and $f(x)$, then (2.4) becomes a rewrite of (2.3).

Proposition 2.4.4 indicates that constructing the confusion matrix is a calibration procedure. Thus, the predictor constructed with constructed using confusion matrix is calibrated and suitable for application with MLLS. [Vaicenavicius et al. (2019)] For any function g, $f(x) = p_s(y|g(x))$ is a calibrated predictor.

We can now summarize the relationship between BBSE and MLLS: A label shift estimator involves two design choices: (i) designing the latent space $p(z|x)$ (which is equivalent to designing a calibrated predictor); and (ii) performing distribution matching in the new space $\mathcal{Z}$. In BBSE, we design a calibrated predictor via the confusion matrix and then perform distribution matching by directly solving linear equations. In general, MLLS does not specify how to obtain a calibrated predictor, but specifies KL minimization as the distribution matching procedure. One can apply the confusion matrix approach to obtain a calibrated predictor and then plug it into MLLS, which is the BBSE analog under MLLS, and is a special case of MLLS.

2.5 Theoretical Analysis of MLLS

We now analyze the performance of MLLS estimator. Even when w^* is the unique optimizer of (2.4) for some calibrated predictor f, assuming convex optimization can be done perfectly, there are still two sources of error preventing us from exactly computing w^* in practice. First, we are optimizing a sample-based approximation (2.5) to the objective in expectation (2.4). We call this source of error *finite-sample error*. Second, the predictor f we use may not be perfectly calibrated on the source distribution as we only have access to samples from source data distribution $p_s(x, y)$. We call this source of error *miscalibration error*. We will first analyze how these two sources of errors affect the estimate of w^* separately and then give a general error bound that incorporates both. All proofs are relegated to Appendix A.5.

Before presenting our analysis, we introduce some notation and regularity assumptions. For any predictor $f : \mathcal{X} \mapsto \Delta^{k-1}$, we define w_f and $\widehat{w}_f$ as in (2.4) and (2.5). If f satisfies the conditions in Theorem 2.4.3 (calibration and linear independence) then we have that $w_f = w^*$. Our goal is to bound $\|\widehat{w}_f - w^*\|$ for a given (possibly miscalibrated) predictor f. We now introduce a regularity condition:

Condition 2.5.1 (Regularity condition for a predictor f)**.** *For any x within the support of $p_t(x)$, i.e. $p_t(x) > 0$, we have both $f(x)^T w_f \geqslant \tau$, $f(x)^T w^* \geqslant \tau$ for some universal constant*

[3] Here, by a random predictor we mean that the predictor outputs a random vector from Δ^{k-1}, not $\mathcal{Y}$.

$\tau > 0$.

Condition 2.5.1 is mild if f is calibrated since in this case $w_f = w^*$ is the maximizer of $\mathbb{E}_t\left[\log f(x)^T w\right]$, and the condition is satisfied if the expectation is finite. Since $f(x)^T w^*$ and $f(x)^T w_f$ are upper-bounded (they are the inner products of two vectors which sum to 1), they also must be lower-bounded away from 0 with arbitrarily high probability without any assumptions. For miscalibrated f, a similar justification holds for assumption that $f(x)^T w_f$ is lower bounded. Turning our attention to the assumption that $f(x)^T w^*$ is lower bounded, we note that it is sufficient if f is close (pointwise) to some calibrated predictor. This in turn is a reasonable assumption on the actual predictor we use for MLLS in practice as it is post-hoc calibrated on source data samples.

Define $\sigma_{f,w}$ to be the minimum eigenvalue of the Hessian $-\nabla_w^2 \mathcal{L}(w, f)$. To state our results compactly we use standard stochastic order notation (see, for instance, (van der Vaart and Wellner, 1996)). We first bound the estimation error introduced by only having finite samples from the target distribution in Lemma 2.5.2. Next, we bound the estimation error introduced by having a miscalibrated f in Lemma 2.5.3.

Lemma 2.5.2. *For any predictor f that satisfies Condition 2.5.1, we have* $\|w_f - \widehat{w}_f\| \leqslant \sigma_{f,w_f}^{-1} \mathcal{O}_p\left(m^{-1/2}\right)$.

Lemma 2.5.3. *For any predictor f and any calibrated predictor f_c that satisfies Condition 2.5.1, we have* $\|w_f - w^*\| \leqslant \sigma_{f,w^*}^{-1} \cdot C \cdot \mathbb{E}_t\left[\|f - f_c\|\right]$ *, for some constant C.*

If we set $f_c(x) = p_s(y|f(x))$, which is a calibrated predictor (Proposition 2.4.4), we can bound the error in terms of the calibration error of f on the source data [4]*:* $\|w_f - w^*\| \leqslant \sigma_{f,w^*}^{-1} \cdot C \cdot \mathcal{E}(f)$.

Note that since $p_s(y) > 0$ for all y, we can upper-bound the error in Lemma 2.5.3 with calibration error on the source data. We combine the two sources of error to bound the estimation error $\|\widehat{w}_f - w^*\|$:

Theorem 2.5.4. *For any predictor f that satisfies Condition 2.5.1, we have*

$$\|\widehat{w}_f - w^*\| \leqslant \sigma_{f,w_f}^{-1} \mathcal{O}_p\left(m^{-1/2}\right) + C \cdot \sigma_{f,w^*}^{-1} \mathcal{E}(f)\,. \tag{2.7}$$

The estimation error of MLLS can be decomposed into (i) finite-sample error, which decays at a rate of $m^{-1/2}$; and (ii) the calibration error of the predictor that we use. The proof is a direct combination of Lemma 2.5.2 and Lemma 2.5.3 applied to the same f with the following error decomposition:

$$\|\widehat{w}_f - w^*\| \leqslant \underbrace{\|w_f - \widehat{w}_f\|}_{\text{finite-sample}} + \underbrace{\|w_f - w^*\|}_{\text{miscalibration}}\,.$$

Theorem 2.5.4 shows that the estimation error depends inversely on the minimum eigenvalue of the Hessian at two different points w_f and w^*. One can unify these two eigenvalues as a single quantity σ_f, the minimum eigenvalue $\mathbb{E}_t\left[f(x)f(x)^T\right]$. We formalize this observation in Appendix A.5.

[4] We present two upper bounds because the second is more interpretable while the first is tighter.

If we use the *post-hoc calibration* procedure (as discussed in Section **??** and A.1) to calibrate a blackbox predictor $\widehat{f}$, we can obtain a bound on the calibration error of f. In more detail, suppose that the class $\mathcal{G}$ used for calibration satisfies standard regularity conditions (injectivity, Lipschitz-continuity, twice differentiability, non-singular Hessian). We have the following lemma:

Lemma 2.5.5. *Let $f = g \circ \widehat{f}$ be the predictor after post-hoc calibration with squared loss l and g belongs to a function class $\mathcal{G}$ that satisfies the standard regularity conditions, we have*

$$\mathcal{E}(f) \leqslant \min_{g \in \mathcal{G}} \mathcal{E}(g \circ \widehat{f}) + \mathcal{O}_p\left(n^{-1/2}\right) \,. \tag{2.8}$$

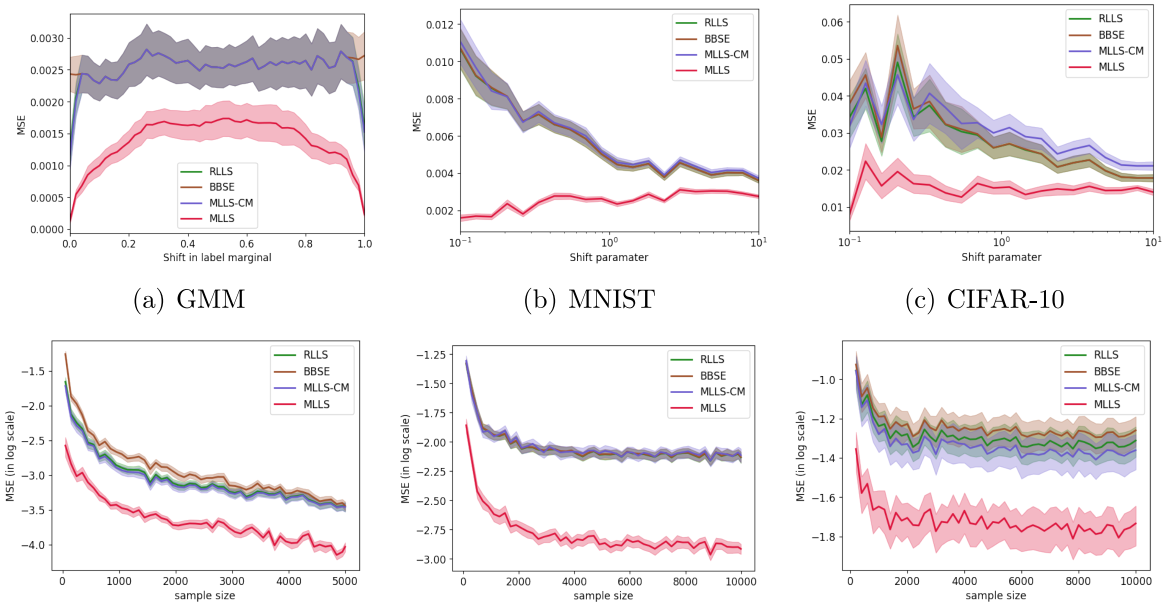
This result is similar to Theorem 4.1 (Kumar et al., 2019). For a model class $\mathcal{G}$ that is rich enough to contain a function $g \in \mathcal{G}$ that achieves zero calibration error, i.e., $\min_{g \in \mathcal{G}} \mathcal{E}(g \circ \widehat{f}) = 0$, then we obtain an estimation error bound for MLLS of $\sigma_f^{-1} \cdot \mathcal{O}_p\left(m^{-1/2} + n^{-1/2}\right)$. This bound is similar to rate of RLLS and BBSE, where instead of σ_f they have minimum eigenvalue of the confusion matrix.

The estimation error bound explains the efficiency of MLLS. Informally, the error of MLLS depends inversely on the minimum eigenvalue of the Hessian of the likelihood σ_f. When we apply coarse calibration via the confusion matrix (in MLLS-CM), we only decrease the value of σ_f. Coarse calibration throws away information (Kuleshov and Liang, 2015) and thus results in greater estimation error for MLLS. In Section 2.6, we empricially show that MLLS-CM's performance is similar to that of BBSE. Moreover, on a synthetic Gaussian mixture model, we show that the minimum eigenvalue of the Hessian obtained using confusion matrix calibration is smaller than the minimum eigenvalue obtained with more granular calibration. Our analysis and observations together suggest MLLS's superior performance than BBSE (or RLLS) is due to the granular calibration but not due to the difference in the optimization objective.

Finally, we want to highlight one minor point regarding applicability of our result. If f is calibrated, Theorem 2.5.4, together with Proposition 3 (in Appendix A.5), implies that MLLS is consistent if $\mathbb{E}_t\left[f(x)f(x)^T\right]$ is invertible. Compared to the consistency condition in Theorem H.2.1 that $\mathbb{E}_s\left[f(x)f(x)^T\right]$ is invertible (together with Proposition 2.4.4), these two conditions are the same if the likelihood ratio $p_t(f(x))/p_s(f(x))$ is lower-bounded. This is true if all entries in w^* are non-zero. Even if w^* contains non-zero entries, the two conditions are still the same if there exists some $w_y^* > 0$ such that $p(f(x)|y)$ covers the full support of $p_s(f(x))$. In general however, the invertibility of $\mathbb{E}_t\left[f(x)f(x)^T\right]$ is a stronger requirement than the invertibility of $\mathbb{E}_s\left[f(x)f(x)^T\right]$. We leave further investigation of this gap for future work.

2.6 Experiments

We experimentally illustrate the performance of MLLS on synthetic data, MNIST (LeCun et al., 1998), and CIFAR10 (Krizhevsky and Hinton, 2009). Following Lipton et al. (2018b), we experiment with *Dirichlet shift* simulations. On each run, we sample a target label



(a) GMM (b) MNIST (c) CIFAR-10

(d) GMM (e) MNIST (f) CIFAR-10

Figure 2.1: (**top**) MSE vs the degree of shift; For GMM, we control the shift in the label marginal for class 1 with a fixed target sample size of 1000. For multiclass problems—-MNIST and CIFAR-10, we control the Dirichlet shift parameter with a fixed sample size of 5000. (**bottom**) MSE (in log scale) vs target sample size; For GMM, we fix the label marginal for class 1 at 0.01 whereas for multiclass problems, MNIST and CIFAR-10, we fix the Dirichlet parameter to 0.1. In all plots, MLLS dominates other methods. All confusion matrix approaches perform similarly, indicating that the advantage of MLLS comes from the choice of calibration but not the way of performing distribution matching.

distribution $p_t(y)$ from a Dirichlet with concentration parameter α. We then generate each target example by first sampling a label $y \sim p_t(y)$ and then sampling (with replacement) an example conditioned on that label . Note that smaller values of alpha correspond to more severe shift. In our experiments, the source label distribution is uniform.

First, we consider a mixture of two Gaussians (as in Example in Section 2.4.3) with $\mu = 1$. With CIFAR10 and MNIST, we split the full training set into two subsets: train and valid, and use the provided test set as is. Then according to the label distribution, we randomly sample with replacement train, valid, and test set from each of their respective pool to form the source and target set. To learn the black box predictor on real datasets, we use the same architecture as Lipton et al. (2018b) for MNIST, and for CIFAR10 we use ResNet-18 (He et al., 2016) as in Azizzadenesheli et al. (2019)[5]. For simulated data, we use the true $p_s(y|x)$ as our predictor function. For each experiment, we sample 100 datasets for each shift parameter and evaluate the empirical MSE and variance of the estimated weights.

[5] We used open source implementation of ResNet-18 https://github.com/kuangliu/pytorch-cifar.

We consider three sets of experiments: (1) MSE vs degree of target shift; (2) MSE vs target sample sizes; and (3) MSE vs calibrated predictors on the source distribution. We refer to MLLS-CM as MLLS with hard confusion matrix calibration as in (2.6). In our experiments, we compare MLLS estimator with BBSE, RLLS, and MLLS-CM. For RLLS and BBSE, we use the publicly available code [6]. To post-hoc calibration, we use BCTS (Alexandari et al., 2021) on the held-out validation set. Using the same validation set, we calculate the confusion matrix for BBSE, RLLS, and MLLS-CM.

We examine the performance of various estimators across all three datasets for various target dataset sizes and shift magnitudes (Figure 2.1). Across all shifts, MLLS (with BCTS-calibrated classifiers) *uniformly dominates* BBSE, RLLS, and MLLS-CM in terms of MSE (Figure 2.1). Observe for severe shifts, MLLS is comparatively dominant. As the available target data increased, all methods improve rapidly, with MLLS outperforming all other methods by a significant margin. Moreover, MLLS's advantages grow more pronounced under extreme shifts. Notice MLLS-CM is roughly equivalent to BBSE across all settings of dataset, target size, and shift magnitude. This concludes MLLS's superior performance is not because of differences in loss function used for distribution matching but due to differences in the granularity of the predictions, caused by crude confusion matrix aggregation.

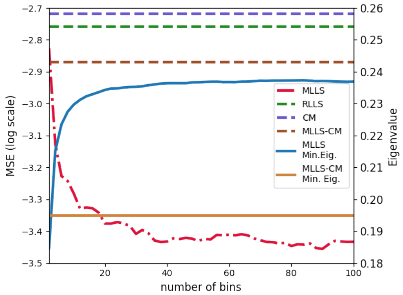


Figure 2.2: MSE (left-axis) with variation of minimum eigenvalue of the Hessian (right-axis) vs number of bins used for aggregation. With increase in number of bins, MSE decrease and the minimum eigenvalue increases.

Note that given a predictor f_1, we can partition our input space and produce another predictor f_2 that, for any data-point gives the expected output of f_1 on points belonging to that partition. If f_1 is calibrated, then f_2 will also be calibrated (Vaicenavicius et al., 2019). On synthetic data, we vary the granularity of calibration (for MLLS) by aggregating $p_s(y|x)$ over a variable number of equal-sized bins. With more bins, less information is lost due to calibration. Consequently, the minimum eigenvalue of the Hessian increases and the MSE decreases, supporting our theoretical bounds (Figure 2.2). We also verify that the confusion matrix calibration performs poorly (Figure 2.2). For MLLS-CM, the minimum eigenvalue of the Hessian is 0.195, significantly smaller than for the binned predictor for $\#\text{bin} \geqslant 4$. Thus, the poor performance of MLLS-CM is predicted by its looser upper bound per our analysis. Note that these experiments presume access to the true predictor $p_s(y|x)$ and thus the MSE strictly improves with the number of bins. In practice, with a fixed source dataset size, increasing the number of bins could lead to overfitting, worsening our calibration.

[6] BBSE: https://github.com/zackchase/label_shift, RLLS: https://github.com/Angela0428/labelshift

2.7 Conclusion

This chapter provides a unified framework relating techniques that use off-the-shelf predictors for label shift estimation. We argue that these methods all employ calibration, either explicitly or implicitly, differing only in the choice of calibration method and their optimization objective. Moreover, with our analysis we show that the choice of calibration method (and not the optimization objective for distribution matching) accounts for the advantage of MLLS with BCTS calibration over BBSE.

In a follow-up work (Roberts et al., 2022), we study the problem of unsupervised learning with multi-domain data under the label shift assumption between domains. This assumption, along with additional assumptions on mixture coefficients (e.g., rank greater than number of classes of the matrix formed by mixture proportion of different classes across domains), allows us to learn a classifier for different classes just from multi-domain data.

Chapter 3

Online Label Shift: Optimal Dynamic Regret meets Practical Algorithms

Based on Baby et al. (2023): Dheeraj Baby*, Saurabh Garg*, Tzu-Ching Yen*, Sivaraman Balakrishnan, Zachary Lipton, and Yu-Xiang Wang. Online label shift: Optimal dynamic regret meets practical algorithms. Advances in Neural Information Processing Systems, 2023

Abstract

In the previous chapter, we assumed setup with static test distribution. This chapter focuses on supervised and unsupervised online label shift, where the class marginals $Q(y)$ varies over time but the class-conditionals $Q(x|y)$ remain invariant. In the unsupervised setting, our goal is to adapt a learner, trained on some offline labeled data, to changing label distributions given unlabeled online data. In the supervised setting, we must both learn a classifier and adapt to the dynamically evolving class marginals given only labeled online data. We develop novel algorithms that reduce the adaptation problem to online regression and guarantee optimal dynamic regret without any prior knowledge of the extent of drift in the label distribution. Our solution is based on bootstrapping the estimates of *online regression oracles* that track the drifting proportions. Experiments across numerous simulated and real-world online label shift scenarios demonstrate the superior performance of our proposed approaches, often achieving 1-3% improvement in accuracy while being sample and computationally efficient. Code is publicly available at this url.

3.1 Introduction

While static source to target distribution shift allows us to systematically study adaptation methods, in the real world, shifts are more likely to occur continually and unpredictably, with data arriving in an *online* fashion. Building on the previous chapter, in this chapter,

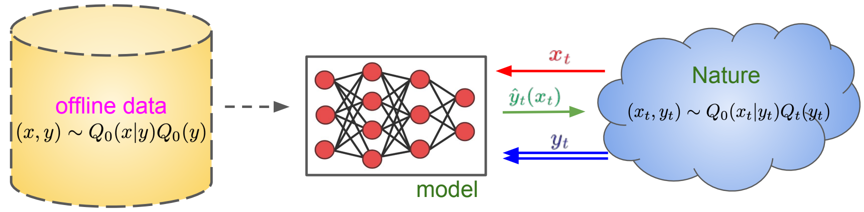

offline data
$(x, y) \sim Q_0(x|y)Q_0(y)$
model
x_t
$\hat{y}_t(x_t)$
y_t
Nature
$(x_t, y_t) \sim Q_0(x_t|y_t)Q_t(y_t)$

Figure 3.1: *UOLS and SOLS setup.* Dashed (double) arrows are exclusive to UOLS (SOLS) settings. Other objects are common to both setups. Central question: how to adapt the model in real-time to drifting label marginals based on all the available data so far?

we will allow temporal shifts in the test label marginal. Beyond the static label shift setting, here we will face one additional challenge: since the label marginal can shift over time, we assume access to limited number of unlabeled examples in each time step. In particular, we will work in setting where we order 10–100 examples from each time-step. While naively incorporating label shift estimation techniques would yield unbiased estimates, these estimates individually will suffer from high variance. Hence, in this chapter, we will explore how mild assumptions on the nature of shift in the label marginal can allows us to obtain minimax estimates.

Researchers have only begun to explore the role that structures like label shift might play in such online settings. Initial attempts to learn under unsupervised online label shifts were made by Wu et al. (2021) and Bai et al. (2022), both of which rely on reductions to Online Convex Optimization (OCO) (Hazan, 2016; Orabona, 2019). This line of research aims in updating a classification model based on online data so that the overall regret is controlled. However, Wu et al. (2021) only control for *static regret* against a fixed classifier (or model) in hindsight and makes the assumption of the convexity (of losses), which is often violated in practice. In the face of online label shift, where the class marginals can vary across rounds, a more fitting notion is to control the *dynamic regret* against a sequence of models in hindsight. Motivated by this observation, Bai et al. (2022) control for the dynamic regret. However, their approach is based on updating model parameters (of the classifier) with online gradient descent and relying on convex losses limits the applicability of their methods (e.g. algorithms in Bai et al. (2022) can not be employed with decision tree classifiers).

In this chapter, we study the problem of learning classifiers under Online Label Shift (OLS) in both *supervised* and *unsupervised* settings (Fig.3.1). In both these settings, the distribution shifts are an online process that respects the label shift assumption. Our primary goal is to develop algorithms that side-step convexity assumptions and at the same time *optimally* adapt to the non-stationarity in the label drift. In the Unsupervised Online Label Shift (UOLS) problem, the learner is provided with a pool of labeled offline data sampled iid from the distribution $Q_0(x, y)$ to train an initial model f_0. Afterwards, at every online round t, few *unlabeled* data points sampled from $Q_t(x)$ are presented. The goal is to adapt f_0 to the non-stationary target distributions $Q_t(x, y)$ so that we can accurately classify the unlabelled data. By contrast, in Supervised Online Label Shift (SOLS), our goal is to learn classifiers from *only* the (labeled) samples that arrive in an online fashion from $Q_t(x, y)$ at each time step, while simultaneously adapting to the non-stationarity induced due to changing label proportions. While SOLS is similar to online learning under non-stationarity, UOLS differs from classical online learning as the test label is not seen during online adaptation. Below are the list of contributions of this chapter.

- **Unsupervised adaptation.** For the UOLS problem, we provide a reduction to online regression (see Defn. 3.2.1), and develop algorithms for adapting the initial classifier f_0 in a computationally efficient way leading to *minimax optimal* dynamic regret. Our approach achieves the best-of-both worlds of Bai et al. (2022); Wu et al. (2021) by controlling the dynamic regret while allowing us to use expressive black-box models for classification (Sec. 3.3).

- **Supervised adaptation.** We develop algorithms for SOLS problem that lead to *minimax optimal* dynamic regret without assuming convexity of losses (Sec. 3.4). Our theoretically optimal solution is based on weighted Empirical Risk Minimization (wERM) with weights tracked by online regression. Motivated by our theory, we also propose a *simple continual learning* baseline which achieves empirical performance competitive to the wERM from scratch at each time step across several semi-synthetic SOLS problems while being 15× more efficient in computation cost.

- **Low switching regressors.** We propose a black-box reduction method to convert an optimal online regression algorithm into another algorithm that switches decisions *sparingly* while *maintaining minimax optimality.* This method is relevant for online change point detection. We demonstrate its application in developing SOLS algorithms to train models only when significant distribution drift is detected, while maintaining statistical optimality (App. B.4 and Algorithm 13).

- **Extensive empirical study.** We corroborate our theoretical findings with experiments across numerous simulated and real-world OLS scenarios spanning vision and language datasets (Sec. 3.5). Our proposed algorithms often improve over the best alternatives in terms of both final accuracy and label marginal estimation. This advantage is particularly prominent with limited initial holdout data (in the UOLS problem) highlighting the *sample efficiency* of our approach.

Even-though online regression is a well studied technique, to the best of our knowledge, it is not used before to address the problem of online label shift. It is precisely the usage of regression which lead to tractable adaptation algorithms while side-stepping convexity assumptions thereby allowing us to use very flexible models for classification. This is in stark contrast to OCO based reductions in (Wu et al., 2021) and (Bai et al., 2022). We propose new theoretical frameworks and identify the right set of assumptions for materializing the reduction to online regression. It was not evident initially that this link would lead to *minimax optimal* dynamic regret rates as well as *consistent* empirical improvement over prior works. Proof of the lower bounds requires adapting the ideas from non-stationary stochastic optimization (Besbes et al., 2015) in a non-trivial manner. Further, none of the proposed methods require the prior knowledge of the extent of distribution drift.

3.2 Problem Setup

Let $\mathcal{X} \subseteq \mathbb{R}^d$ be the input space and $\mathcal{Y} = [K] := \{1, 2, \ldots, K\}$ be the output space. Let Q be a distribution over $\mathcal{X} \times \mathcal{Y}$ and let $q(\cdot)$ denotes the corresponding label marginal. Δ_K is the K-dimensional simplex. For a vector $v \in \mathbb{R}^K$, $v[i]$ is its i^{th} coordinate. We assume that we have a hypothesis class $\mathcal{H}$. For a function $f \in \mathcal{H} : \mathcal{X} \to \Delta_K$, we also use $f(i|x)$ to indicate $f(x)[i]$. With $\ell(f(x), y)$, we denote the loss of making a prediction with the classifier f on (x, y). L denotes the expected loss, i.e., $L = \mathbb{E}_{(x,y)\sim Q}\left[\ell(f(x), y)\right]$. $\widetilde{O}(\cdot)$ hides dependencies in absolute constants and poly-logarithmic factors of horizon and failure probabilities.

In this work, we study online learning under distribution shift, where the distribution $Q_t(x, y)$ may continuously change with time. Simialr to the previous chapter, we focus on the *label shift* assumption where the distribution over label proportions $q_t(y)$ can change arbitrarily but the distribution of the covariate conditioned on a label value (i.e., $Q_t(x|y)$) is assumed to be invariant across all time steps. We refer to this setting as Online Label Shift (OLS). Here, we consider settings of unsupervised and supervised OLS settings captured in Frameworks 1 and 3 respectively. In both settings, at round t a sample (x_t, y_t) is drawn from a distribution with density $Q_t(x_t, y_t)$. In the UOLS setting, the label is not revealed to the learner. However, we assume access to offline labeled data sampled iid from Q_0 which we use to train an initial classifier f_0. The goal is to adapt the initial classifier f_0 to drifting label distributions. In contrast, for the SOLS setting, the label is revealed to the learner after making a prediction and the goal is to learn a classifier $f_t \in \mathcal{H}$ for each time step.

Next, we formally define the concept of online regression which will be central to our discussions. Simply put, an online regression algorithm tracks a ground truth sequence from noisy observations.

Definition 3.2.1 (online regression)**.** *Fix any $T > 0$. The following interaction scheme is defined to be the online regression protocol.*

- *At round $t \in [T]$, an algorithm predicts $\widehat{\theta}_t \in \mathbb{R}^K$.*
- *A noisy version of ground truth $z_t = \theta_t + \epsilon_t$ is revealed where $\theta_t, \epsilon_t \in \mathbb{R}^K$, and $\|\epsilon_t\|_2, \|\theta_t\|_2 \leqslant B$. Further the noise ϵ_t are independent across time with $E[\epsilon_t] = 0$ and $\mathrm{Var}(\epsilon_t[i]) \leqslant \sigma^2\ \forall i \in [K]$.*

An online regression algorithm aims to control $\sum_{t=1}^{T} \|\widehat{\theta}_t - \theta_t\|_2^2$. Moreover, the regression algorithm is defined to be adaptively minimax optimal if with probability at least $1 - \delta$, $\sum_{t=1}^{n} \|\widehat{\theta}_t - \theta_t\|_2^2 = \widetilde{O}(T^{1/3} V_T^{2/3})$ without knowing V_T ahead of time. Here $V_T := \sum_{t=2}^{T} \|\theta_t - \theta_{t-1}\|_1$ is termed as the Total Variation (TV) of the sequence $\theta_{1:T}$.

3.3 Unsupervised Online Label Shift

In this section, we develop a framework for handling the UOLS problem. We summarize the setup in Framework 1. Since in practice, we may need to work with classifiers such as deep neural networks or decision trees, we do not impose convexity assumptions on

Framework 1 Unsupervised Online Label Shift (UOLS) protocol

Input: Initial classifier $f_0 : \mathcal{X} \to \Delta_K$ trained on offline labeled dataset $\{(x_i, y_i)\}_{i=1}^N$ sampled iid from Q_0;

```
$f_1 = f_0$
for each round $t \in [T]$ do
    Nature samples $x_t \in \mathcal{X}$ and $y_t \in \mathcal{Y}$, with $(x_t, y_t) \sim Q_t$; Only $x_t$ is revealed to the learner.
    Learner predicts a label $i \sim f_t(x_t) \in \Delta_K$.
    $f_{t+1} = \mathcal{A}(f_0, x_{1:t})$, where $\mathcal{A}$ is strategy to adapt the classifier based on past data.
end for
```

Algorithm 2 `RegressAndReweight` to handle UOLS

Input: i) Online regression oracle ALG; ii) Initial classifier f_0; iii) The confusion matrix C; iv) The label marginal $q_0 \in \mathcal{D}$ of the training distribution;

```
At round $t$, get the classifier covariate $x_t$.
Let $\widehat{w}q_t = \Pi_{\mathcal{D}}(\mathrm{ALG}(s_{1:t-1}))$, where $\Pi_{\mathcal{D}}(x) = \arg\min_{y \in \mathcal{D}} \|y - x\|_2$.
Sample a label $i$ with probability $\propto \frac{\widehat{w}q_t(i)}{q_0(i)} f_0(i|x_t)$.
Let $s_t = C^{-1} f_0(x_t)$.
Update the online regression oracle with the estimate $s_t$.
```

the (population) loss of the classifier as a function of the model parameters. Despite the absence of such simplifying assumptions, we provide performance guarantees for our label shift adaption techniques so that they are certified to be fail-safe.

Under the label shift assumption, we have $Q_t(y|x)$ as a re-weighted version of $Q_0(y|x)$:

$$Q_t(y|x) = \frac{Q_t(y)}{Q_t(x)} Q_t(x|y) = \frac{Q_t(y)}{Q_t(x)} Q_0(x|y) = \frac{Q_t(y) Q_0(x)}{Q_t(x) Q_0(y)} Q_0(y|x) \propto \frac{Q_t(y)}{Q_0(y)} Q_0(y|x), \tag{3.1}$$

where the second equality is due to the label shift assumption. Hence, a reasonable strategy is to re-weight the initial classifier f_0 with label proportions (estimate) at the current step, since we only have to correct the label distribution shift. This re-weighting technique is widely used for offline label shift correction (Alexandari et al., 2021; Azizzadenesheli et al., 2019; Lipton et al., 2018a) and for learning under label imbalance (Cui et al., 2019; Huang et al., 2016; Wang et al., 2017b).

Our starting point in developing a framework is inspired by Bai et al. (2022); Wu et al. (2021) . For self-containedness, we briefly recap their arguments next. We refer interested readers to their papers for more details. Wu et al. (2021) considers a hypothesis class of re-weighted initial classifier f_0. The loss of a hypothesis is parameterised by the re-weighting vector. They use tools from OCO to optimise the loss and converge to a best fixed classifier. However as noted in Wu et al. (2021), the losses are not convex with respect to the re-weight vector in practice. Hence usage of OCO techniques is not fully satisfactory in their problem formulation.

In a complementary direction, Bai et al. (2022) abandons the idea of re-weighting. Instead, they update the parameters of a model at each round using online gradient descent and a loss function whose expected value is assumed to be convex with respect to model parameters. They provide dynamic regret guarantees against a sequence of changing model parameters in hindsight, and connects it to the variation of the true label marginals. More precisely, they provide algorithms with $\sum_{t=1}^{T} L_t(w_t) - L_t(w_t^*)$ to be well controlled where w_t^* is the best model parameter to be used at round t and L_t is a (population level) loss function. However, there are some scopes for improvement in this direction as well. For example, the convexity assumption can be easily violated when working with interpretable models based on decision trees, or if we want to retrain few final layers of a deep classifier based on new data. Further as noted in the experiments (Sec. 2.6), their methods based on retraining the classifier require more data than re-weighting based methods. Our experiments also indicate that re-weighting can be computationally cheaper than re-training without sacrificing the classifier accuracy.

Thus, on the one hand, the work of Wu et al. (2021) allows us to use the power of expressive initial classifiers while only controlling the static regret against a fixed hypothesis. On the other hand, the work of Bai et al. (2022) allows controlling the dynamic regret while limiting the flexibility of deployed models. We next provide our framework for handling label shifts that achieves the best of both worlds by controlling the dynamic regret while allowing the use of expressive *blackbox* models.

In summary, we estimate the sequence of online label marginals and leverage the idea of re-weighting an initial classifier as in Wu et al. (2021). In particular, given an estimate $\widehat{w}q_t(y)$ of the true label marginal at round t, we compute the output of the re-weighted classifier f_t as $\frac{\widehat{w}q_t(y)}{q_0(y)} f_0(y|x) / Z$ where $Z = \sum_y \frac{\widehat{w}q_t(y)}{q_0(y)} f_0(y|x)$. However, to get around the issue of non-convexity, we separate out the process of estimating the re-weighting vectors via a reduction to online regression which is a well-defined and convex problem with computationally efficient off-the-shelf algorithms readily available. Second, and more importantly, Wu et al. (2021) competes with the best *fixed* re-weighted hypothesis. However, in the problem setting of label shift, the true label marginals are in fact changing. Hence, we control the *dynamic regret* against a sequence of re-weighted hypotheses in hindsight. All proofs for the next sub-section are deferred to App. B.3.

3.3.1 Proposed algorithm and performance guarantees

We start by presenting our assumptions. This is followed by the main algorithm for UOLS and its performance guarantees. Similar to the treatment in Bai et al. (2022), we assume the following.

Assumption 1. *Assume access to the true label marginals $q_0 \in \Delta_K$ of the offline training data and the true confusion matrix $C \in \mathbb{R}^{K \times K}$ with $C_{ij} = E_{x \sim Q_0(\cdot|y=j),} f_0(i|x)$. Further the minimum singular value $\sigma_{min}(C) = \Omega(1)$ is bounded away from zero.*

As noted in prior work (Garg et al., 2020a; Lipton et al., 2018a), the invertibility of the confusion matrix holds whenever the classifier f_0 has good accuracy and the true label marginal q_0 assigns a non-zero probability to each label. Though we assume perfect knowledge of the label marginals of the training data and the associated confusion matrix, this restriction can be easily relaxed to their empirical counterparts computable from the training data. The finite sample error between the empirical and population quantities can be bounded by $O(1/\sqrt{N})$ where N is the number of initial training data samples. To this end, we operate in the regime where the time horizon obeys $T = O(\sqrt{N})$. However, similar to Bai et al. (2022), we make this assumption mainly to simplify presentation without trivializing any aspect of the OLS problem.

Next, we present our assumptions on the loss function. Let $p \in \Delta_K$. Consider a classifier that predicts a label $\widehat{w}y(x)$, by sampling $\widehat{w}y(x)$ according to the distribution that assigns a weight $\frac{p(i)}{q_0(i)} f_0(i|x)$ to the label i. Define $L_t(p)$ to be any non-negative loss that ascertains the quality of the marginal p. For example, $L_t(p) = E[\ell(\widehat{y}(x), y)]$ where the expectation is taken wrt the randomness in the draw $(x, y) \sim Q_t$ and in sampling $\widehat{y}(x)$. Here ℓ is any classification loss (e.g. 0-1, cross-entropy).

Assumption 2 (Lipschitzness of loss functions)**.** *Let $\mathcal{D}$ be a compact and convex domain. Assume that $L_t(p)$ is G Lipschitz with $p \in \mathcal{D} \subseteq \Delta_K$, i.e, $L_t(p_1) - L_t(p_2) \leqslant G\|p_1 - p_2\|_2$ for any $p_1, p_2 \in \mathcal{D}$. The constant G need not be known ahead of time.*

We show in Lemmas B.3.1 and B.3.2 that the above assumption is satisfied under mild regularity conditions. Furthermore, the prior works such as Wu et al. (2021) and Bai et al. (2022) also require that losses are Lipschitz with a *known* Lipschitz constant apriori to set the step sizes for their OGD based methods.

The main goal here is to design appropriate re-weighting estimates such that the *dynamic regret*:

$$R_{\text{dynamic}}(T) = \sum_{t=1}^{T} L_t(\widehat{q}_t) - L_t(q_t) \leqslant \sum_{t=1}^{T} G\|\widehat{q}_t - q_t\|_2 \tag{3.2}$$

is controlled where $\widehat{q}_t \in \Delta_K$ is the estimate of the true label marginal q_t. Thus we have reduced the problem of handling OLS to the problem of online estimation of the true label marginals.

Under label shift, we can get an unbiased estimate of the true marginals at any round via the techniques in Alexandari et al. (2021); Azizzadenesheli et al. (2019); Lipton et al. (2018a). More precisely, $s_t = C^{-1} f_0(x_t)$ has the property that $E[s_t] = q_t$ (see Lemma B.3.3). Further, the variance of the estimate s_t is bounded by $1/\sigma^2_{min}(C)$. Unfortunately, these unbiased estimates can not be directly used to track the moving marginals q_t. This is because the total squared error $\sum_{t=1}^{T} E[\|s_t - q_t\|_2^2]$ grows linearly in T as the sum of the variance of the point-wise estimates accumulates unfavorably over time.

To get around these issues, one can use online regression algorithms such as FLH (Hazan and Seshadhri, 2007) with online averaging base learners or the Aligator algorithm (Baby et al., 2021). These algorithms use ensemble methods to (roughly) output running averages of s_t where the variation in the *true* label marginals is small enough. The averaging within intervals where the true marginals change slowly helps to reduce the overall variance while injecting only a small bias. We use such *online regression oracles* to track the moving marginals and re-calibrate the initial classifier. Overall, Algorithm 2 summarizes our method which has the following performance guarantee.

Theorem 3.3.1. *Suppose we run Algorithm 2 with the online regression oracle ALG as FLH-FTL (App. B.6) or Aligator (Baby et al., 2021). Then under Assumptions 1 and 2, we have*

$$E[R_{dynamic}(T)] = \widetilde{O}\left(\frac{K^{1/6}T^{2/3}V_T^{1/3}}{\sigma_{min}^{2/3}(C)} + \frac{\sqrt{KT}}{\sigma_{min}(C)}\right), \tag{3.3}$$

where $V_T := \sum_{t=2}^{T} \|q_t - q_{t-1}\|_1$ and the expectation is taken with respect to randomness in the revealed co-variates. Further, this result is attained without prior knowledge of V_T.

Remark 3.3.1. *We emphasize that any valid online regression oracle ALG can be plugged into Algorithm 2. This implies that one can even use transformer-based time series models to track the moving marginals q_t. Further, we have the flexibility of choosing the initial classifier to be any* black-box *model that outputs a distribution over the labels.*

Remark 3.3.2. *Unlike prior works such as (Bai et al., 2022; Wu et al., 2021), we do not need a pre-specified bound on the gradient of the losses. Consequently Eq.(3.2) holds for the smallest value of the Lipschitzness coefficient G, leading to tight regret bounds. Further, the projection step in Line 2 of Algorithm 2 is done only to safeguard our theory against pathological scenarios with unbounded Lipschitz constant for losses. In our experiments, we do not perform such projections.*

We next show that the performance guarantee in Theorem 3.3.1 is optimal (modulo factors of $\log T$) in a minimax sense.

Theorem 3.3.2. *Let $V_T \leqslant 64T$. There exists a loss function, a domain $\mathcal{D}$ (in Assumption 2), and a choice of adversarial strategy for generating the data such that for any algorithm, we have $\sum_{t=1}^{T} E([L_t(\widehat{q}_t)] - L_t(q_t)) = \Omega\left(\max\{T^{2/3}V_T^{1/3}, \sqrt{T}\}\right)$, where $\widehat{q}_t \in \mathcal{D}$ is the weight estimated by the algorithm and $q_t \in \mathcal{D}$ is the label marginal at round t chosen by the adversary. Here the expectation is taken with respect to the randomness in the algorithm and the adversary.*

3.4 Supervised Online Label Shift

In this section, we focus on the SOLS problem where the labels are revealed to the learner after it makes decisions. Framework 3 summarizes our setup. Let $f_t^* := \arg\min_{f \in \mathcal{H}} L_t(f)$ be the population minimiser. We aim to control the *dynamic regret* against the best sequence of hypotheses in hindsight:

$$R^{\mathcal{H}}_{\text{dynamic}}(T) =: \sum_{t=1}^{T} L_t(f_t) - L_t(f_t^*)\,. \tag{3.5}$$

If the SOLS problem is convex, it reduces to OCO (Hazan, 2016; Orabona, 2019) and existing works provide $\widetilde{O}(T^{2/3}V_T^{1/3})$ dynamic regret guarantees (Zhang et al., 2018b). However, in

Framework 3 Supervised Online Label Shift (SOLS) protocol

```
input A hypothesis class $\mathcal{H}$.
for each round $t \in [T]$ do
    Nature samples $N$ iid data points $x_{t,1:N} \in \mathcal{X}$ and $y_{t,1:N} \in \mathcal{Y}$, with each $(x_{t,i}, y_{t,i}) \sim Q_t$; $x_{t,1:N}$ is revealed to the learner.
    For each $i \in [N]$, learner predicts a label $f_t(x_{t,i})$.
    The label $y_{t,i} \in \mathcal{Y}$ for each $i \in [N]$ is revealed.
    $f_{t+1} = \mathcal{A}(f_t, \{x_{1:t,1:N}, y_{1:t,1:N}\})$ where algorithm $\mathcal{A}$ updates the classifier with past data.
end for
```

Algorithm 4 `TrainByWeights` to handle SOLS

```
input Online regression oracle ALG, hypothesis class $\mathcal{H}$
At round $t \in [T]$, get estimated label marginal $\widehat{q}_t$ from ALG$(s_{1:t-1})$.
Update the hypothesis with weighted ERM:
    $$f_t = \arg\min_{f \in \mathcal{H}} \sum_{i=1}^{t-1} \sum_{j=1}^{N} \frac{\widehat{q}_t(y_{i,j})}{\widehat{q}_i(y_{i,j})} \ell(f(x_{i,j}), y_{i,j}) \quad (3.4)$$
Get co-variates $x_{t,1:N}$ and make predictions with $f_t$
Get labels $y_{t,1:N}$
Compute $s_t[i] = \frac{1}{N} \sum_{j=1}^{N} \mathbb{I}\{y_{t,j} = i\}$ for all $i \in [K]$.
Update ALG with the empirical label marginals $s_t$.
```

practice, since loss functions are seldom convex with respect to model parameters in modern machine learning, the performance bounds of OCO algorithms cease to hold true. In our work, we extend the generalization guarantees of ERM from statistical learning theory (Bousquet et al., 2003) to the SOLS problem. All proofs of next sub-section are deferred to App. B.5.

3.4.1 Proposed algorithms and performance guarantees

We start by providing a simple initial algorithm whose computational complexity and flexibility will be improved later. Note that due to the label shift assumption, for any $j, t \in [T]$, we have $E_{(x,y)\sim Q_t}[\ell(f(x), y)] = E_{(x,y)\sim Q_j}\left[\frac{q_t(y)}{q_j(y)}\ell(f(x), y)\right]$. Here we assume that the true label marginals $q_t(y) > 0$ for all $t \in [T]$ and all $y \in [K]$. Based on this, we propose a simple weighted ERM approach (Algorithm 4) where we use an online regression oracle to estimate the label marginals from the (noisy) empirical label marginals computed with observed labeled data. With weighted ERM and plug-in estimates of importance weights, we can obtain our classifier f_t. One can expect that by adequately choosing the online regression oracle ALG, the risk of the hypothesis f_t computed will be close to that of f_t^*. Here the degree of closeness will also depend on the number of data points seen thus far. Consequently, Algorithm 4 controls the dynamic regret (Eq.(3.5)) in a graceful manner. We have the following performance guarantee:

Theorem 3.4.1. *Suppose the true label marginal satisfies* $\min_{t,k} q_t(k) \geqslant \mu > 0$. *Choose the online regression oracle in Algorithm 4 as FLH-FTL (App. G.3) or Aligator from Baby et al. (2021) with its predictions clipped such that* $\widehat{q}_t[k] \geqslant \mu$. *Then with probability at least*

$1-\delta$, Algorithm 4 produces hypotheses with $R^{\mathcal{H}}_{\text{dynamic}} = \tilde{O}\left(T^{2/3}V_T^{1/3} + \sqrt{T\log(|\mathcal{H}|/\delta)}\right)$, where $V_T = \sum_{t=2}^{T} \|q_t - q_{t-1}\|_1$. Further, this result is attained without any prior knowledge of the variation budget V_T.

The above rate contains the sum of two terms. The second term is the familiar rate seen in the supervised statistical learning theory literature under iid data (Bousquet et al., 2003). The first term reflects the price we pay for adapting to distributional drift in the label marginals. While we prove this result for finite hypothesis sets, the extension to infinite sets is direct by standard covering net arguments (Vershynin, 2018).

Remark 3.4.1. *Theorem 3.4.1 requires that the estimates of the label marginals to be clipped from below by μ. This is done only to facilitate theoretical guarantees by enforcing that the importance weights used in Eq.(8.1) do not become unbounded. However, note that only the labels we actually observe enters the objective in Eq.(8.1). In particular, if a label has very low probability of getting sampled at a round, then it is unlikely that it enters the objective. Due to this reason, in our experiments, we haven't used the clipping operation (see Section 2.6 and Appendix G.3 for more details).*

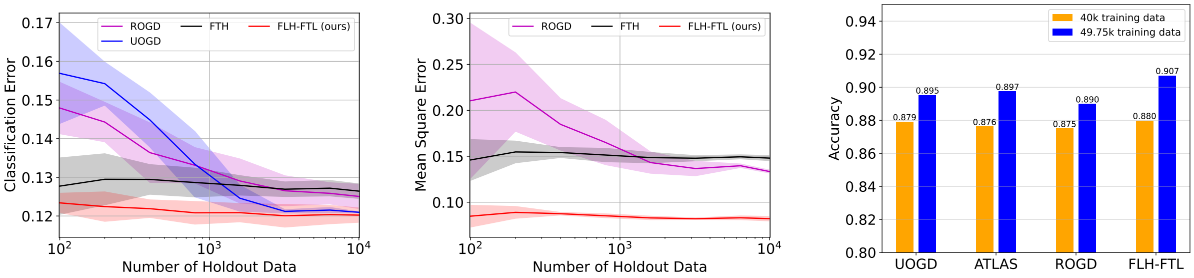
The proof of the theorem uses concentration arguments to establish that the risk of the hypothesis f_t is close to the risk of the optimal f_t^*. However, unlike the standard offline supervised setting with iid data, for any fixed hypothesis, the terms in the summation of Eq.(8.1) are correlated through the estimates of the online regression oracle. We handle it by introducing uncorrelated surrogate random variables and bounding the associated discrepancy. Next, we show (near) minimax optimality of the guarantee in Theorem 3.4.1.

Theorem 3.4.2. *Let $V_T \leqslant T/8$. There exists a choice of hypothesis class, loss function, and adversarial strategy of generating the data such that $R^{\mathcal{H}}_{\text{dynamic}} = \Omega\left(T^{2/3}V_T^{1/3} + \sqrt{T\log(|\mathcal{H}|)}\right)$, where the expectation is taken with respect to randomness in the algorithm and adversary.*

Remark 3.4.2. *Though the rates in Theorems 3.3.2 and 3.4.2 are similar, we note that the corresponding regret definitions are different. Hence the minimax rates are not directly comparable between the supervised and unsupervised settings.*

Even-though Algorithm 4 has attractive performance guarantees, it requires retraining with weighted ERM at every round. This can be computationally expensive. To alleviate this issue, we design a new online change point detection algorithm (Algorithm 12 in App. B.4) that can adaptively discover time intervals where the label marginals change slow enough. We show that the new online change point detection algorithm can be used to significantly reduce the number of retraining steps without sacrificing statistical efficiency (up to constants). Due to space constraints, we defer the exact details to App. B.4. We remark that our change point detection algorithm is applicable to general online regression problems and hence can be of independent interest to online learning community.

Remark 3.4.3. *Algorithm 12 helps to reduce the run-time complexity. However, both Algorithms 4 and 12 have the drawback of storing all data points accumulated over the online rounds. This is reminiscent to FTL / FTRL type algorithms from online learning.*



(a) (b) (c)

Figure 3.2: *Results on the UOLS problem.* **(a) and (b):** Ablation on CIFAR10 with monotone shift over sizes of holdout data used to update model parameters and compute confusion matrix, with amount of training data held fixed. FLH-FTL (ours) outperforms all other alternatives throughout in classification error and mean square error in label marginal estimation. Unlike the alternatives, the performance of FLH-FTL (ours) is unaffected by the decrease in amount of holdout data. **(c):** CIFAR10 results with monotone shift using varying amount of training data, with the remaining labeled data used as holdout (total number of samples fixed to 50k). The performance of FLH-FTL is minimally impacted by the reduction in the quantity of holdout data, thus yielding the greatest advantage from utilizing a larger volume of training data.

We leave the task of deriving theoretical guarantees with reduced storage complexity under non-convex losses as an important future direction.

3.5 Experiments[1]

3.5.1 UOLS Setup and Results

Setup Following the dataset setup of Bai et al. (2022), we conducted experiments on synthetic and common benchmark data such as MNIST (LeCun et al., 1998), CIFAR-10 (Krizhevsky and Hinton, 2009), Fashion (Xiao et al., 2017), EuroSAT (Helber et al., 2019), Arxiv (Clement et al., 2019), and SHL (Gjoreski et al., 2018; Wang et al., 2019c). For each dataset, the original data is split into labeled data available during offline training and validation, and the unlabeled data that we observe during online learning. We experiment with varying sizes of holdout offline data which is used to obtain the confusion matrix and update the model parameters to adapt to OLS to probe the sample efficiency of all the methods. In contrast to previous works (Bai et al., 2022; Wu et al., 2021), we have chosen to use a smaller amount of holdout offline data for our main experiments. We made this decision because the standard practice for deployment involves training and validating models on training and holdout splits, respectively (e.g., with k-fold cross-validation). Then, the final model is deployed by training on all available data (i.e., the union of train and holdout) with the identified hyperparameters. However, to employ UOLS techniques in

[1] Code is publicly available at https://github.com/Anon-djiwh/OnlineLabelShift.

Methods	Synthetic		MNIST		CIFAR		EuroSAT		Fashion		ArXiv	
	Ber	Sin	Ber	Sin	Ber	Sin	Ber	Sin	Ber	Sin	Ber	Sin
Base	8.6±0.2	8.2±0.3	4.9±0.4	3.9±0.0	16±0	16±0	13±0	13±0	15±0	15±0	23±1	19±0
OFC	6.4±0.6	5.5±0.2	4.4±0.5	3.2±0.3	12±1	11±0	11±1	10±1	7.9±0.1	7.1±0.1	20±2	15±0
Oracle	3.7±0.8	3.9±0.2	2.5±0.5	1.5±0.1	5.4±0.5	5.8±0.1	3.9±0.3	4.1±0.1	3.7±0.2	3.6±0.1	7.7±1.0	5.1±0.1
FTH	6.5±0.6	5.7±0.3	4.5±0.6	**3.3±0.2**	11±0	**11±0**	10±0	**9.6±0.0**	8.5±0.3	**6.9±0.4**	20±1	**14±0**
FTFWH	6.6±0.5	5.7±0.3	4.5±0.6	**3.3±0.2**	11±1	**11±0**	9.8±0.4	**9.6±0.1**	8.2±0.6	**6.9±0.4**	20±1	**14±0**
ROGD	7.9±0.3	7.2±0.6	6.2±2.8	4.4±1.5	16±3	13±0	14±1	13±1	10±1	8.2±0.7	23±2	17±1
UOGD	8.1±0.6	7.5±0.6	5.4±0.6	4.0±0.0	14±0	14±1	10±1	9.8±0.7	11±2	11±2	21±1	17±1
ATLAS	8.0±1.0	7.5±0.6	5.2±0.6	3.7±0.2	13±0	13±1	10±1	9.9±0.7	12±2	12±2	21±1	16±0
FLH-FTL (ours)	**5.4±0.7**	**5.4±0.4**	**4.4±0.7**	**3.3±0.2**	**10±0**	**11±0**	**9.2±0.4**	**9.6±0.1**	**7.7±0.4**	7.0±0.0	**19±1**	**14±0**

	Synthetic		MNIST		CIFAR		EuroSAT		Fashion		ArXiv	
	Ber	Sin	Ber	Sin	Ber	Sin	Ber	Sin	Ber	Sin	Ber	Sin
FTH	0.19±0.01	0.10±0.00	0.27±0.00	0.14±0.00	0.27±0.01	0.14±0.00	0.27±0.00	0.14±0.00	0.29±0.01	**0.14±0.01**	0.29±0.01	**0.15±0.00**
FTFWH	0.19±0.02	0.09±0.00	0.26±0.02	0.13±0.00	0.25±0.02	**0.13±0.00**	0.25±0.01	**0.13±0.00**	0.25±0.04	**0.14±0.01**	0.27±0.02	**0.15±0.00**
ROGD	0.29±0.03	0.24±0.01	0.41±0.08	0.37±0.06	0.39±0.04	0.30±0.05	0.43±0.04	0.35±0.03	0.37±0.02	0.30±0.01	0.34±0.03	0.28±0.01
FLH-FTL (ours)	**0.10±0.01**	**0.08±0.00**	**0.15±0.01**	**0.12±0.00**	**0.17±0.01**	**0.13±0.00**	**0.16±0.01**	**0.13±0.00**	**0.18±0.02**	**0.14±0.01**	**0.23±0.01**	**0.15±0.00**

Table 3.1: *Results for UOLS problems under sinusoidal (Sin) and Bernoulli (Ber) shifts.* **Top:** Classification Error. **Bottom:** Mean-squared error in estimating label marginal. For both, lower is better. Across all datasets, we observe that FLH-FTL (ours) often improves over best alternatives.

practice, practitioners must hold out data that was not seen during training to update the model during online adaptation. Therefore, methods that are efficient with respect to the amount of offline holdout data required might be preferable.

For all datasets except SHL, we simulate online label shifts with four types of shifts studied in Bai et al. (2022): monotone shift, square shift, sinusoidal shift, and Bernoulli shift. For SHL locomotion, we use the real-world shift occurring over time. For architectures, we use an MLP for Fashion, SHL and MNIST, Resnets (He et al., 2016) for EuroSAT, CINIC, and CIFAR, and DistilBERT (Sanh et al., 2019b; Wolf et al., 2019) based models for arXiv. For alternate approaches, along with a base classifier (which does no adaptation) and oracle classifier (which reweight using the true label marginals), we make comparisons with adaptation algorithms proposed in prior works (Bai et al., 2022; Wu et al., 2021). In particular, we compare with ROGD, FTH, FTFWH from Wu et al. (2021) and UOGD, ATLAS from Bai et al. (2022). For brevity, we refer to our method as FLH-FTL (though strictly speaking, our methods are based on FLH from Hazan and Seshadhri (2007) with online averages as base learners). We run all the online label shift experiments with the time horizon $T = 1000$ and at each step 10 samples are revealed. We repeat all experiments

	Base	Oracle	ROGD	FTH	FTFWH	FLH-FTL (ours)
Cl Err	18±1	6.3±1.3	19±3	14±2	14±2	**13±2**
MSE	NA	0.0±0.0	0.3±0.0	0.3±0.0	0.3±0.0	**0.2**±0.0

Table 3.2: *Results with a Random Forest classifier on MNIST dataset.* Note that methods that update model parameters are not applicable here. FLH-FTL outperforms existing alternatives for both accuracy and label marginal estimation.

	CT (base)	CT-RS (ours) w FTH	CT-RS (ours) w FLH-FTL	w-ERM (oracle)
Cl Err	20.0±0.5	18.38±0.4	**17.12±0.8**	16.32±0.7
MSE	NA	0.18±0.01	**0.12±0.01**	NA

Table 3.3: *Results on SOLS setup* on CIFAR10 SOLS with Bernoulli shift. CT with RS improves over the base model (CT) and achieves competitive performance with respect to weighted ERM oracle. MNIST results are similar (see App. B.6).

with 3 seeds to obtain means and standard deviations of the results. For other methods that perform re-weighting correction on softmax predictions, we use the labeled holdout data to calibrate the model with temperature scaling, which tunes one temperature parameter (Guo et al., 2017). We provide exact details about the datasets, label shift simulations, models, and prior methods in App. B.6.

Results Overall, across all datasets, we observe that our method FLH-FTL performs better than alternative approaches in terms of both classification error and mean squared error for estimating the label marginal. Note that methods that directly update the model parameters (i.e., UOGD, ATLAS) do not provide any estimate of the label marginal (Table 3.1). UOGD and ATLAS also require offline holdout labeled data (i.e., from time step 0) to make online updates to the model parameters. For this purpose, we use the same labeled data that we use to compute the confusion matrix.

As we increase the holdout offline labeled dataset size for updating the model parameters (and to compute the confusion matrix), we observe that classification error and MSE with FLH-FTL stay (relatively) constant whereas the classification errors of other alternatives improve (Fig. 3.2). This highlights that FLH-FTL can be much more sample efficient with respect to the size of the hold-out offline labeled data. Motivated by this observation, we perform an additional experiment in which we increase the offline training data and observe that we can overall improve the classification accuracy significantly with FLH-FTL (Fig. 3.2). We present results on SHL dataset with similar findings on semi-synthetic datasets in App. B.6.2. Finally, we also experiment with a random forest model on the MNIST dataset. Note methods that update model parameters (e.g., UOGD and ATLAS) with OGD are not applicable here. Here, we also observe that we improve over existing applicable alternatives (Table 3.2).

3.5.2 SOLS setup and results

Setup For the supervised problem, we experiment with MNIST and CIFAR datasets. We simulate a time horizon of $T = 200$. For each dataset, at each step, we observe 50 samples with Bernoulli shift. Motivated by our theoretical results with weighted ERM, we propose a simple baseline which continually trains the model at every step instead of starting ERM from scratch every time. We maintain a pool of all the labeled data received till that time step, and at every step, we randomly sample a batch with uniform label marginal to update the model. Finally, we re-weight the updated softmax outputs with estimated label marginal. We call this method Continual Training via Re-Sampling (CT-RS). Its relation as a close variant of weighted ERM is elaborated in App. B.6.1. To estimate the label marginal, we try FTH and ours FLH-FTL.

Results On both datasets, we observe that empirical performance with CT-RS improves over the naive continual training baseline. Additionally, CT-RS results are competitive with weighted ERM while being 5–15× faster in terms of computation cost (we include the exact computational cost in App. B.6.1). Moreover, as in UOLS setup, we observe that FLH-FTL improves over FTH for both target label marginal estimation and classification.

3.6 Conclusion

In this work, we focused on unsupervised and supervised online label shift settings. For both settings, we developed algorithms with minimax optimal dynamic regret. Experimental results on both real and semi-synthetic datasets substantiate that our methods improve over prior works both in terms of accuracy and target label marginal estimation.

Part II

Adaptation Under Input Distribution Shift and Relaxed Label Shift Scenarios

Chapter 4

Mixture Proportion Estimation and PU Learning: A Modern Approach

Based on Garg et al. (2021b): Saurabh Garg, Yifan Wu, Alexander J. Smola, Sivaraman Balakrishnan, and Zachary Lipton. Mixture proportion estimation and PU learning: A modern approach.Advances in Neural Information Processing Systems, 2021

Abstract

In the next two chapters, we relax the label shift assumption to allow previously unseen classes. This chapter deals with the base case which is classically studied under the paradigm of Positive and Unlabeled (PU) learning.

Given only positive examples and unlabeled examples (containing both positive and negative classes), the problem can be broken into two subtasks: (i) *Mixture Proportion Estimation* (MPE)—determining the fraction of positive examples in the unlabeled data; and (ii) *PU-learning*—given such an estimate, learning the desired positive-versus-negative classifier. Unfortunately, classical methods for both problems break down in high-dimensional settings. Meanwhile, recently proposed heuristics lack theoretical coherence and depend precariously on hyperparameter tuning. We propose two simple techniques: *Best Bin Estimation* (BBE) for MPE; and *Conditional Value Ignoring Risk* (CVIR), for PU-learning. Both methods dominate previous approaches empirically, and for BBE, we establish formal guarantees that hold whenever we can train a model to cleanly separate out a small subset of positive examples. Our final algorithm (TED)^n, alternates between the two procedures, significantly improving both our mixture proportion estimator and classifier. Code is available at this url.

4.1 Introduction

In previous chapters, we focused on problems where the classes in the test unlabeled data were a subset of training data classes. While this allows us to develop principled machinery to tackle problems where additional classes are not introduced in test data, it remains ineffective in scenarios where previously unseen classes appear in unlabeled test data. In this and the next chapter, we will introduce algorithms and identifiablity conditions to handle such scenarios.

When deploying k-way classifiers in the wild, what can we do when confronted with data from a previously unseen class $(k+1)$? Theory dictates that learning under distribution shift is impossible absent assumptions. And yet people appear to exhibit this capability routinely. Faced with new surprising symptoms, doctors can recognize the presence of a previously unseen ailment and attempt to estimate its prevalence. Similarly, naturalists can discover new species, estimate their range and population, and recognize them reliably going forward.

To begin making this problem tractable, we might make the label shift assumption (Lipton et al., 2018b; Saerens et al., 2002; Storkey, 2009), i.e., that while the class balance $p(y)$ can change, the class conditional distributions $p(x|y)$ do not. Moreover, we might begin by focusing on the base case, where only one class has been seen previously, i.e., $k = 1$. Here, we possess (labeled) positive data from the source distribution, and (unlabeled) data from the target distribution, consisting of both positive and negative instances. This problem has been studied in the literature as *learning from positive and unlabeled data* (De Comité et al., 1999; Letouzey et al., 2000) and has typically been broken down into two subtasks: (i) Mixture Proportion Estimation (MPE) where we estimate α, the fraction of positives among the unlabeled examples; and (ii) PU-learning where this estimate is incorporated into a scheme for learning a Positive-versus-Negative (PvN) binary classifier.

Traditionally, MPE and PU-learning have been motivated by settings involving large databases where unlabeled examples are abundant and a small fraction of the total positives have been extracted. For example, medical records might be annotated indicating the presence of certain diagnoses, but the unmarked passages are not necessarily negative. This setup has also been motivated by protein and gene identification (Elkan and Noto, 2008). Databases in molecular biology often contain lists of molecules known to exhibit some characteristic of interest. However, many other molecules may exhibit the desired characteristic, even if this remains unknown to science.

Many methods have been proposed for both MPE (Bekker and Davis, 2018; Du Plessis and Sugiyama, 2014a; Elkan and Noto, 2008; Ivanov, 2019; Jain et al., 2016; Ramaswamy et al., 2016; Reeve and Kabán, 2019; Scott, 2015) and PU-learning (Du Plessis et al., 2015; 2014; Kiryo et al., 2017). However, classical MPE methods break down in high-dimensional settings (Ramaswamy et al., 2016) or yield estimators whose accuracy depends on restrictive conditions (Du Plessis and Sugiyama, 2014a; Scott, 2015). On the other hand, most recent proposals either lack theoretical coherence, rely on heroic assumptions, or depend precariously on tuning hyperparameters that are, by the very problem setting, untunable.

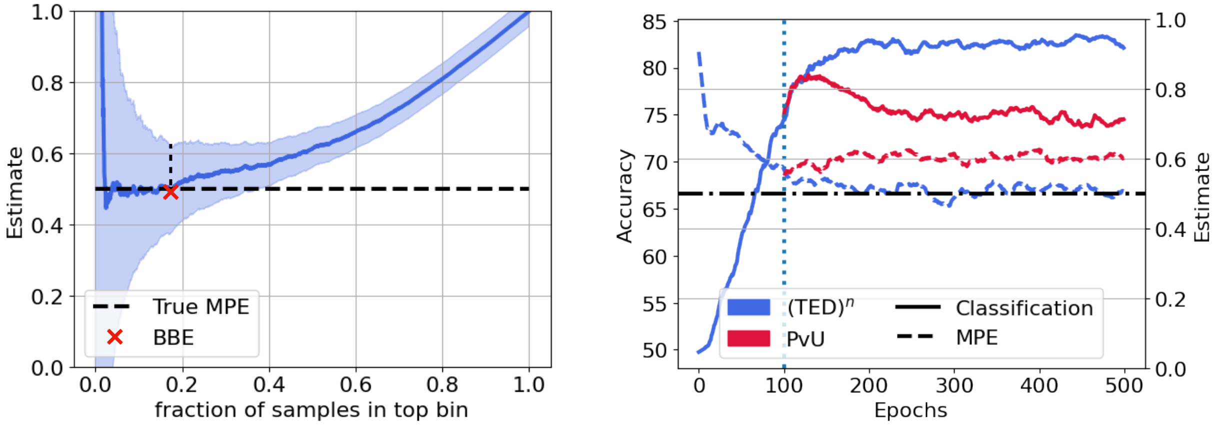


Figure 4.1: *Illustration of proposed methods.* **(left)** Estimate of α with varying fraction of unlabeled examples in the top bin. The shaded region highlights the upper and lower confidence bounds. BBE selects the top bin that minimizes the upper confidence bound. **(right)** Accuracy and MPE estimate as training proceeds. Till 100-th epoch (vertical line), we perform PvU training, i.e., warm start for (TED)^n. Post 100-th epoch, we continue with both (TED)^n and PvU training. Note that (TED)^n improves both classification accuracy and MPE compared to PvU training. Results with Resnet-18 on binary-CIFAR. For details and comparisons with other methods, see Sec. 4.6.

For PU learning, Elkan and Noto (2008) suggest training a classifier to distinguish positive from unlabeled data followed by a rescaling procedure. Du Plessis et al. (2015) suggest an unbiased risk estimation framework for PU learning. However, these methods fail badly when applied with model classes capable of overfitting and thus implementations on high-dimensional datasets rely on extensive hyperparameter tuning and additional ad-hoc heuristics that do not transport effectively across datasets.

In this chapter, we propose (i) Best Bin Estimation (BBE), an effective technique for MPE that produces consistent estimates $\widehat{\alpha}$ under mild assumptions and admits finite-sample statistical guarantees achieving the desired $O(1/\sqrt{n})$ rates; and (ii) learning with the Conditional Value Ignoring Risk (CVIR) objective, which discards the highest loss $\widehat{\alpha}$ fraction of examples on each training epoch, removing the incentive to overfit to the unlabeled positive examples. Both methods are simple to implement, compatible with arbitrary hypothesis classes (including deep networks), and dominate existing methods in our experimental evaluation. Finally, we combine the two in an iterated Transform-Estimate-Discard (TED)^n framework that significantly improves both MPE estimation error and classifier error.

We build on label shift methods (Alexandari et al., 2021; Azizzadenesheli et al., 2019; Garg et al., 2020b; Lipton et al., 2018b; Rabanser et al., 2019), that leverage black-box classifiers to reduce dimensionality, estimating the target label distribution as a functional of source and target push-forward distributions. While label shift methods rely on classifiers trained to separate previously seen classes, BBE is able to exploit a Positive-versus-Unlabeled

(PvU) target classifier, which gives each input a score indicating how likely it is to be a positive sample. In particular, BBE identifies a threshold such that by estimating the ratio between the fractions of positive and unlabeled points receiving scores above the threshold, we obtain the mixture proportion α.

BBE works because in practice, for many datasets, PvU classifiers, even when uncalibrated, produce outputs with near monotonic calibration diagrams. Higher scores correspond to a higher proportion of positives, and when the positive data contains a separable sub-domain, i.e., a region of the input space where only the positive distribution has support, classifiers often exhibit a threshold above which the *top bin* contains mostly positive examples. We show that the existence of a (nearly) pure top bin is sufficient for BBE to produce a (nearly) consistent estimate $\widehat{\alpha}$, whose finite sample convergence rates depend on the fraction of examples in the bin and whose bias depends on the *purity* of the bin. Crucially, we can estimate the optimal threshold from data.

We conduct a battery of experiments both to empirically validate our claim that BBE's assumptions are mild and frequently hold in practice, and to establish the outperformance of BBE, CVIR, and $(\text{TED})^n$ over the previous state of the art. We first motivate BBE by demonstrating that in practice PvU classifiers tend to isolate a reasonably large, reasonably pure top bin. We then conduct extensive experiments on semi-synthetic data, adapting a variety of binary classification datasets to the PU learning setup and demonstrating the superior performance of BBE and PU-learning with the CVIR objective. Moreover, we show that $(\text{TED})^n$, which combines the two in an iterative fashion, improves significantly over previous methods across several architectures on a range of image and text datasets.

4.2 Related Work

Research on MPE and PU learning date to (De Comité et al., 1999; Denis, 1998; Letouzey et al., 2000) (see review by (Bekker and Davis, 2020)). Elkan and Noto (2008) first proposed to leverage a PvU classifier to estimate the mixture proportion. Du Plessis and Sugiyama (2014b) propose a different method for estimating the mixture coefficient based on Pearson divergence minimization. While they do not require a PvU classifier, they suffer the same shortcoming. Both methods require that the positive and negative examples have disjoint support. Our requirements are considerably milder. Blanchard et al. (2010) observe that without assumptions on the underlying positive and negative distributions, the mixture proportion is not identifiable. Furthermore, (Blanchard et al., 2010) provide an *irreducibility* condition that identifies α and propose an estimator that converges to the true α. While their estimator can converge arbitrarily slowly, Scott (2015) showed faster convergence ($\mathcal{O}(1/\sqrt{n})$) under stronger conditions. Unfortunately, despite its appealing theoretical properties Blanchard et al. (2010)'s estimator is computationally infeasible. Building on Blanchard et al. (2010), Sanderson and Scott (2014) and Scott (2015) proposed estimating the mixture proportion from a ROC curve constructed for the PvU classifier. However, when the PvU classifier is not perfect, these methods are not clearly understood. Ramaswamy et al. (2016) proposed the first computationally feasible algorithm for MPE with convergence guarantees to the true proportion. Their method KM, requires embedding distributions onto an RKHS. However, their estimator underperforms on high dimensional datasets and scales poorly with large datasets. Bekker and Davis (2018) proposed TIcE, hoping to identify a positive subdomain in the input space using decision tree induction. This method also underperforms in high-dimensional settings.

In the most similar works, Jain et al. (2016) and Ivanov (2019) explore dimensionality reduction using a PvU classifier. Both methods estimate α through a procedure operating on the PvU classifier's output. However, neither methods has provided theoretical backing. (Ivanov, 2019) concede that their method often fails and returns a zero estimate, requiring that they fall back to a different estimator. Moreover while both papers state that their method require the Bayes-optimal PvU classifier to identify α in the transformed space, we prove that even when hypothesis class is well specified for PvN learning, PvU training can fail to recover the Bayes-optimal scoring function. Furthermore, we also show that the heuristic estimator in Scott (2015) can be thought of as using PvU classifier for dimensionality reduction. While this heuristic is similar to our estimator in spirit, we show that the functional form of their estimator is different from ours and note that their heuristic enjoys no theoretical guarantee. By contrast, our estimator BBE is theoretically coherent under mild conditions and outperforms all of these methods empirically.

Given α, Elkan and Noto (2008) propose a transformation via Bayes rule to obtain the PvN classifier. They also propose a weighted objective, with weights given by the PvU classifier. Other propose unbiased risk estimators (Du Plessis et al., 2015; 2014) which require the mixture proportion α. Du Plessis et al. (2014) proposed an unbiased estimator with non-convex loss functions satisfying a specific symmetric condition, and subsequently Du Plessis et al. (2015) generalized it to convex loss functions (denoted uPU in our experiments). in our experiments. Noting the problem of overfitting in modern overparameterized models, Kiryo et al. (2017) propose a regularized extension that clips the loss on unlabeled data to zero. This is considered the current state-of-the-art in PU literature (denoted nnPU in our experiments). More recently, Ivanov (2019) proposed DEDPUL, which finetunes the PvU classifiers using several heuristics, Bayes rule, and Expectation Maximization (EM). Since their method only applies a post-processing procedure, they rely on a good domain discriminator classifier in the first place and several hyperparameters for their heuristics. Several classical methods attempt to learn weights that identify reliable negative examples (Lee and Liu, 2003; Li and Liu, 2003; Liu et al., 2002; 2003; Zhang and Lee, 2005). However, these earlier methods have not been successful with modern deep learning models.

4.3 Problem Setup

By $\|\cdot\|$ and $\langle \cdot, \cdot \rangle$, we denote the Euclidean norm and inner product, respectively. For a vector $v \in \mathbb{R}^d$, we use v_j to denote its j^{th} entry, and for an event E, we let $\mathbb{I}[E]$ denote the binary indicator of the event. By $|A|$, we denote the cardinality of set A. Let $\mathcal{X} \in \mathbb{R}^d$ be the input space and $\mathcal{Y} = \{-1, +1\}$ be the output space. Let $\mathrm{P} : \mathcal{X} \times \mathcal{Y} \to [0, 1]$ be the underlying joint distribution and let p denote its corresponding density.

Algorithm 5 Best Bin Estimation (BBE)

```
input : Validation positive ($X_p$) and unlabeled ($X_u$) samples. Blackbox model classifier
    $\widehat{f} : \mathcal{X} \to [0,1]$. Hyperparameter $0 < \delta, \gamma < 1$.
  $Z_p, Z_u = f(X_p), f(X_u)$.
  $\widehat{w}q_p(z), \widehat{w}q_u(z) = \frac{\sum_{z_i \in Z_p} \mathbb{I}[z_i \geqslant z]}{n_p}, \frac{\sum_{z_i \in Z_u} \mathbb{I}[z_i \geqslant z]}{n_u}$ for all $z \in [0,1]$.
  Estimate $\widehat{w}c := \arg\min_{c \in [0,1]} \left( \frac{\widehat{w}q_u(c)}{\widehat{w}q_p(c)} + \frac{1+\gamma}{\widehat{w}q_p(c)} \left( \sqrt{\frac{\log(4/\delta)}{2n_u}} + \sqrt{\frac{\log(4/\delta)}{2n_p}} \right) \right)$.
output : $\widehat{w}\alpha := \frac{\widehat{w}q_u(\widehat{w}c)}{\widehat{w}q_p(\widehat{w}c)}$
```

Let P_{P} and P_{N} be the class-conditional distributions for positive and negative class and $p_p(x) = p(x|y = +1)$ and $p_n(x) = p(x|y = -1)$ be the corresponding class-conditional densities. P_{U} denotes the distribution of the unlabeled data and p_u denotes its density. Let $\alpha \in [0,1]$ be the fraction of positives among the unlabeled population, i.e., $\mathrm{P}_{\mathsf{U}} = \alpha \mathrm{P}_{\mathsf{P}} + (1-\alpha)\mathrm{P}_{\mathsf{N}}$. When learning from positive and unlabeled data, we obtain i.i.d. samples from the positive (class-conditional) distribution, which we denote as $X_p = \{x_1, x_2, \ldots, x_{n_p}\} \sim \mathrm{P}_{\mathsf{P}}^{n_p}$ and i.i.d samples from unlabeled distribution as $X_u = \{x_{n_p+1}, x_{n_p+2}, \ldots, x_{n_p+n_u}\} \sim \mathrm{P}_{\mathsf{U}}^{n_u}$.

MPE is the problem of estimating α. Absent assumptions on P_{P}, P_{N} and P_{U}, the mixture proportion α is not identifiable (Blanchard et al., 2010). Indeed, if $\mathrm{P}_{\mathsf{U}} = \alpha \mathrm{P}_{\mathsf{P}} + (1-\alpha)\mathrm{P}_{\mathsf{N}}$, then any alternate decomposition of the form $\mathrm{P}_{\mathsf{U}} = (\alpha - \gamma)\mathrm{P}_{\mathsf{P}} + (1 - \alpha + \gamma)\mathrm{P}'_{\mathsf{N}}$, for $\gamma \in [0, \alpha)$ and $\mathrm{P}'_{\mathsf{N}} = (1-\alpha+\gamma)^{-1}(\gamma \mathrm{P}_{\mathsf{P}} + (1-\alpha)\mathrm{P}_{\mathsf{N}})$, is also valid. Since we do not observe samples from the distribution P_{N}, the parameter α is not identifiable. Blanchard et al. (2010) formulate an *irreducibility* condition under which α is identifiable. Intuitively, the condition restricts P_{N} to ensure that it can not be a (non-trivial) mixture of P_{P} and any other distribution. While this irreducibility condition makes α identifiable, in the worst-case, the parameter α can be difficult to estimate and any estimator must suffer an arbitrarily slow rate of convergence (Blanchard et al., 2010). In this paper, we propose mild conditions on the PvU classifier that make α identifiable and allows us to derive finite-sample convergence guarantees.

With PU learning, the aim is to learn a classifier $f : \mathcal{X} \to [0,1]$ to approximate $p(y = +1|x)$. We assume that we are given a loss function $\ell : [0,1] \times \mathcal{Y} \to \mathbb{R}$, such that $\ell(z, y)$ is the loss incurred by predicting z when the true label is y. For a classifier f and a sampled set $X = \{x_1, x_2, \ldots, x_n\}$, we let $\widehat{w}L^{+}(f; X) = \sum_{i=1}^{n} \ell(f(x_i), +1)/n$ denote the loss when predicting the samples as positive and $\widehat{w}L^{-}(f; X) = \sum_{i=1}^{n} \ell(f(x_i), -1)/n$ the loss when predicting the samples as negative. For a sample set X each with true label y, we define 0-1 error as $\widehat{w}\mathcal{E}^{y}(f; X) = \sum_{i=1}^{n} \mathbb{I}\left[y(f(x_i) - t) \leqslant 0\right]/n$ for some predefined threshold t . Unless stated otherwise, the threshold is assumed to be 0.5.

4.4 Mixture Proportion Estimation

In this section, we introduce BBE, a new method that leverages a blackbox classifier f to perform MPE and establish convergence guarantees. All proofs are relegated to App. C.1. To begin, we assume access to a fixed classifier f. For intuition, you may think of f as a PvU classifer trained on some portion fo the positive and unlabeled examples. In Sec. 4.5, we discuss other ways to obtain a suitable classifier from PU data.

We now introduce some additional notation. Assume f transforms an input $x \in \mathcal{X}$ to $z \in [0,1]$, i.e., $z = f(x)$. For given probability density function p and a classifier f, define a function $q(z) = \int_{A_z} p(x)dx$, where $A_z = \{x \in \mathcal{X} : f(x) \geqslant z\}$ for all $z \in [0,1]$. Intuitively, $q(z)$ captures the cumulative density of points in a top bin, the proportion of input domain that is assigned a value larger than z by the classifier f in the transformed space. We now define an empirical estimator $\widehat{w}q(z)$ given a set $X = \{x_1, x_2, \ldots, x_n\}$ sampled iid from $p(x)$. Let $Z = f(X)$. Define $\widehat{w}q(z) = \sum_{i=1}^{n} \mathbb{I}\left[z_i \geqslant z\right]/n$. For each pdf p_p, p_n and p_u, we define q_p, q_n and q_u respectively.

Without any assumptions on the underlying distribution and the classifier f, we aim to estimate $\alpha^* = \min_{c \in [0,1]} q_u(c)/q_p(c)$ with BBE. Later, under one mild assumption that empirically holds across numerous PU datasets, we show that $\alpha^* = \alpha$, i.e., α^* matches the true mixture proportion α.

Our procedure proceeds as follows: First, given a held-out dataset of positive (X_p) and unlabeled examples (X_u), we push all examples through the classifier f to obtain one-dimensional outputs $Z_p = f(X_p)$ and $Z_u = f(X_u)$. Next, with Z_p and Z_u, we estimate $\widehat{w}q_p$ and $\widehat{w}q_u$. Finally, we return the ratio $\widehat{w}q_u(\widehat{w}c)/\widehat{w}q_p(\widehat{w}c)$ at $\widehat{w}c$ that minimizes the upper confidence bound (calculated using Lemma D.4.2) at a pre-specified level δ and a fixed parameter $\gamma \in (0,1)$. Our method is summarized in Algorithm 17. For theoretical guarantees, we multiply the confidence bound term with $1+\gamma$ for a small positive constant γ. Refer to App. C.1.1 for details. We now show that the proposed estimator comes with the following guarantee:

Theorem 4.4.1. *Define $c^* = \arg\min_{c \in [0,1]} q_u(c)/q_p(c)$. For $\min(n_p, n_u) \geqslant \frac{2\log(4/\delta)}{q_p(c^*)}$ and for every $\delta > 0$, the mixture proportion estimator $\widehat{w}\alpha$ defined in Algorithm 17 satisfies with probability $1-\delta$:*

$$|\widehat{w}\alpha - \alpha^*| \leqslant \frac{c}{q_p(c^*)} \left(\sqrt{\frac{\log(4/\delta)}{n_u}} + \sqrt{\frac{\log(4/\delta)}{n_p}} \right),$$

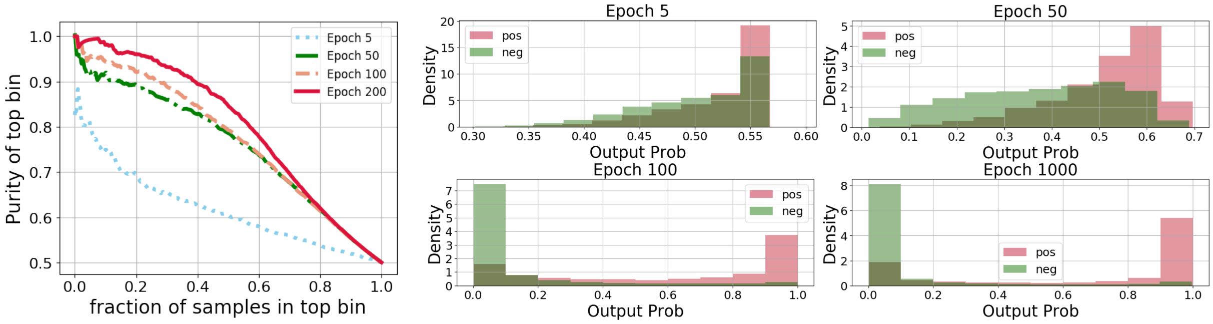
for some constant $c \geqslant 0$.

Theorem 4.4.1 shows that with high probability, our estimate is close to α^*. The proof of the theorem is based on the following confidence bound inequality:

Lemma 4.4.2. *For every $\delta > 0$, with probability at least $1-\delta$, we have for all $c \in [0,1]$*

$$\left| \frac{\widehat{w}q_u(c)}{\widehat{w}q_p(c)} - \frac{q_u(c)}{q_p(c)} \right| \leqslant \frac{1}{\widehat{w}q_p(c)} \left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + \frac{q_u(c)}{q_p(c)} \sqrt{\frac{\log(4/\delta)}{2n_p}} \right).$$

Now, we discuss the convergence of our estimator to the true mixture proportion α. Since, $p_u(x) = \alpha p_p(x) + (1-\alpha)p_n(x)$, for all $x \in \mathcal{X}$, we have $q_u(z) = \alpha q_p(z) + (1-\alpha)q_n(z)$, for all $z \in [0,1]$.



(a) (b)

Figure 4.2: (a) Purity and size (in terms of fraction of unlabeled samples) in the top bin and (b) Distribution of predicted probabilities (of being positive) for unlabeled training data as training proceeds with (TED)^n. Results with ResNet-18 on binary-CIFAR. As in Fig. 11.1, we fix W at 100. In App. C.7.3, we show that as PvU training proceeds, the purity of top bin degrades and the distribution of predicted probabilities of positives and negatives become less and less separable.

Corollary 4.4.3. *Define* $c^* = \arg\min_{c \in [0,1]} q_n(c)/q_p(c)$. *Assume* $\min(n_p, n_u) \geqslant \frac{2\log(4/\delta)}{q_p(c^*)}$. *For every* $\delta > 0$, $\widehat{w}\alpha$ *(in Algorithm 17) satisfies with probability* $1 - \delta$*:*

$$\alpha - \frac{c_1}{q_p(c^*)}\left(\sqrt{\frac{\log(4/\delta)}{n_u}} + \sqrt{\frac{\log(4/\delta)}{n_p}}\right) \leqslant \widehat{w}\alpha\,, \textit{ and}$$

$$\widehat{w}\alpha \leqslant \alpha + (1-\alpha)\frac{q_n(c^*)}{q_p(c^*)} + \frac{c_2}{q_p(c^*)}\left(\sqrt{\frac{\log(4/\delta)}{n_u}} + \sqrt{\frac{\log(4/\delta)}{n_p}}\right)\,,$$

for some constant $c_1, c_2 \geqslant 0$*.*

As a corollary to Theorem 4.4.1, we show that our estimator $\widehat{w}\alpha$ converges to the true α with convergence rate $\min(n_p, n_u)^{-1/2}$, as long as there exist a threshold $c_f \in (0,1)$ such that $q_p(c_f) \geqslant \epsilon_p$ and $q_n(c_f) = 0$ for some constant $\epsilon_p > 0$. We refer to this condition as the *pure positive bin* property.

Note that in a more general case, our bound in Corollary 4.4.3 captures the tradeoff due to the proportion of negative examples in the top bin (bias) versus the proportion of positives in the top bin (variance).

Empirical Validation We now empirically validate the positive pure top bin property (Fig. 4.2). We observe that as PvU training proceeds, purity of the top bin improves for a fixed fraction of samples in the top bin. Moreover, this behavior becomes more pronounced when learning a PvU classifier with the CVIR objective proposed in the following section.

Comparison with existing methods Due to the intractability of Blanchard et al. (2010) estimator, Scott (2015) implements a heuristic based on identifying a point on

Algorithm 6 PU learning with Conditional Value Ignoring Risk (CVIR) objective

```
input : Labeled positive training data ($X_p$) and unlabeled training samples ($X_u$). Mixture
    proportion estimate $\alpha$.
Initialize a training model $f_\theta$ and an stochastic optimization algorithm $\mathcal{A}$.
$X_n := X_u$.
while training error $\widehat{w}\mathcal{E}^+(f_\theta; X_p) + \widehat{w}\mathcal{E}^-(f_\theta; X_n)$ is not converged do
    Rank samples $x_u \in X_u$ according to their loss values $\ell(f_\theta(x_u), -1)$.
    $X_n := X_{u,1-\alpha}$ where $X_{u,1-\alpha}$ denote the lowest ranked $1 - \alpha$ fraction of samples.
    Shuffle $(X_p, X_n)$ into $B$ mini-batches. With $(X_p^i, X_n^i)$ we denote $i$-th mini-batch.
    for $i = 1$ to $B$ do
        Set the gradient $\nabla_\theta \left[\alpha \cdot \widehat{w}L^+(f_\theta; X_p^i) + (1 - \alpha) \cdot \widehat{w}L^-(f_\theta; X_n^i)\right]$ and update $\theta$ with
            algo. $\mathcal{A}$.
    end for
end while
output : Trained classifier $f_\theta$
```

the AUC curve such that the slope of the line segment between this point and (1,1) is minimized. While this approach is similar in spirit to our BBE method, there are some striking differences. First, the heuristic estimator in Scott (2015) provides no theoretical guarantees, whereas we provide guarantees that BBE will converge to the best estimate achievable over all choices of the bin size and provide consistent estimates whenever a pure top bin exists. Second, while both estimates involve thresholds, the functional form of the estimates are different. Corroborating theoretical results of BBE, we observe that the choices in BBE create substantial differences in the empirical performance as observed in App. C.2. We work out details of comparison between Scott (2015) heuristic and BBE in App. C.2.

On the other hand, recent works (Ivanov, 2019; Jain et al., 2016) that use PvU classifier for dimensionality reduction, discuss Bayes optimality of the PvU classifier (or its one-to-one mapping) as a sufficient condition to preserve α in transformed space. By contrast, we show that the milder pure positive bin property is sufficient to guarantee consistency and achieve $\mathcal{O}(1/\sqrt{n})$ rates. Furthermore, in a simple toy setup in App. C.3, we show that even when the hypothesis class is well specified for PvN learning, it will not in general contain the Bayes optimal PvU classifier and thus PvU training will not recover the Bayes-optimal scoring function, even in population. Contrarily, we show that any monotonic mapping of the Bayes-optimal PvU scoring function induces a positive pure top bin property. We leave further theoretical investigations concerning conditions under which a pure positive top bin arises to future work.

4.5 PU-Learning

Given positive and unlabeled data, we hope not only to identify α, but also to obtain a classifier that distinguishes effectively between positive and negative samples. In supervised learning with separable data (e.g., cleanly labeled image data), overparameterized models generalize well even after achieving near-zero training error. However, with PvU training over-parameterized models can memorize the unlabeled positives, assigning them confidently to the negative class, which can severely hurt generalization on PN data (Zhang et al., 2017). Moreover, while unbiased losses exist that estimate the PvN loss given PU data and the mixture proportion α, this unbiasedness only holds before the loss is optimized, and becomes ineffective with powerful deep learning models capable of memorization.

A variety of heuristics, including ad-hoc early stopping criteria, have been explored (Ivanov, 2019), where training proceeds until the loss on unseen PU data ceases to decrease. However, this approach leads to severe under-fitting (results in App. C.7.2). On the other hand, by regularizing the loss function, nnPU Kiryo et al. (2017) mitigates overfitting issues due to memorization.

However, we observe that nnPU still leaves a substantial accuracy gap when compared to a model trained just on the positive and negative (from the unlabeled) data (ref. experiment in App. C.7.1). This leads us to ask the following question: *can we improve performance over nnPU of a model just trained with PU data and bridge this gap?* In an ideal scenario, if we could identify and remove all the positive points from the unlabeled data during training then we can hope to achieve improved performance over nnPU. Indeed, in practice, we observe that in the initial stages of PvU training, the model assigns much higher scores to positives than to negatives in the unlabeled data (Fig. D.1(b)).

Inspired by this observation, we propose CVIR, a simple yet effective objective for PU learning. Below, we present our method assuming an access to the true MPE. Later, we combine BBE with CVIR optimization, yielding (TED)^n, an alternating optimization that significantly improves both the BBE estimates and the PvU classifier.

Given a training set of positives X_p and unlabeled X_u and the mixture proportion α, we begin by ranking the unlabeled data according the predicted probability (of being positive) by our classifier. Then, in every epoch of training, we create a (temporary) set of provisionally negative samples X_n by removing α fraction of the unlabeled samples currently scored as most positive. Next, we update our classifier by minimize the loss on the positives X_p and provisional negatives X_n by treating them as negatives. We repeat this procedure until the training error on X_p and X_n converges. Likewise nnPU, note that this procedure does not need early stopping. Summary in Algorithm 6.

In App. C.4, we justify our loss function in the scenario when the positives and negatives are separable. For a more general scenario, we show that each step of our alternating procedure in CVIR cannot increase the population loss and hence, CVIR can only improve (or plateau) after every iteration.

(TED)^n Integrating BBE and CVIR We are now ready to present our algorithm Transfer, Estimate and Discard (TED)^n that combines BBE and CVIR objective.

First, we observe the interaction between BBE and CVIR objective. If we have an accurate mixture proportion estimate, then it leads to improved classifier, in particular, we reject

Algorithm 7 Transform-Estimate-Discard $(\text{TED})^n$

```
input : Positive data ($X_p$) and unlabeled samples ($X_u$). Hyperparameter $W, \delta$.
1: Initialize a training model $f_\theta$ and an stochastic optimization algorithm $\mathcal{A}$.
2: Randomly split positive and unlabeled data into training $X_p^1, X_u^1$ and hold-out set ($X_p^2, X_u^2$).
3: $X_n^1 := X_u^1$.
   {// Warm start with domain discrimination training}
4: for $i = 1$ to $W$ do
5:     Shuffle ($X_p^1, X_n^1$) into $B$ mini-batches. With ($X_p^{1^i}, X_n^{1^i}$) we denote $i$-th mini-batch.
6:     for $i = 1$ to $B$ do
7:         Set the gradient $\nabla_\theta \left[ \widehat{w} L^+(f_\theta; X_p^{1^i}) + \widehat{w} L^-(f_\theta; X_n^{1^i}) \right]$ and update $\theta$ with algorithm $\mathcal{A}$.
8:     end for
9: end for
10: while training error $\widehat{w}\mathcal{E}^+(f_\theta; X_p^1) + \widehat{w}\mathcal{E}^-(f_\theta; X_n^1)$ is not converged do
11:     Estimate $\widehat{w}\alpha$ using Algorithm 17 with ($X_p^2, X_u^2$) and $f_\theta$ as input.
12:     Rank samples $x_u \in X_u^1$ according to their loss values $l(f_\theta(x_u), -1)$.
13:     $X_n^1 := X_{u,1-\widehat{w}\alpha}^1$ where $X_{u,1-\widehat{w}\alpha}^1$ denote the lowest ranked $1 - \widehat{w}\alpha$ fraction of samples.
14:     Train model $f_\theta$ for one epoch on ($X_p^1, X_n^1$) as in Lines 4-7.
15: end while
output : Trained classifier $f_\theta$
```

accurate number of prospective positive samples from unlabeled. Consequently, updating the classifier to minimize loss on positive versus retained unlabeled improves purity of top bin. This leads to an obvious alternating procedure where at each epoch, we first use BBE to estimate $\widehat{w}\alpha$ and then update the classifier with CVIR objective with $\widehat{w}\alpha$ as input. We repeat this until training error has not converged. Our method is summarized in Algorithm 7.

Note that we need to warm start with PvU (positive versus negative) training, since in the initial stages mixture proportion estimate is often close to 1 rejecting all the unlabeled examples. However, in next section, we show that our procedure is not sensitive to the choice of number of warm start epochs and in a few cases with large datasets, we can even get away without warm start (i.e., $W = 0$) without hurting the performance. Moreover, recall that our aim is to distinguish positive versus negative examples among the unlabeled set where the proportion of positives is determined by the true mixture proportion α. However, unlike CVIR, we do not re-weight the losses in $(\text{TED})^n$. While true MPE α is unknown, one natural choice is to use the estimate $\widehat{w}\alpha$ at each iteration. However, in our initial experiments, we observed that re-weighted objective with estimate $\widehat{w}\alpha$ led to comparatively poor classification performance due to presence of bias in estimate $\widehat{w}\alpha$ in the initial iterations. We note that for deep neural networks (for which model mis-specification is seldom a prominent concern) and when the underlying classes are separable (as with most image datasets), it is known that importance weighting has little to no effect on the final

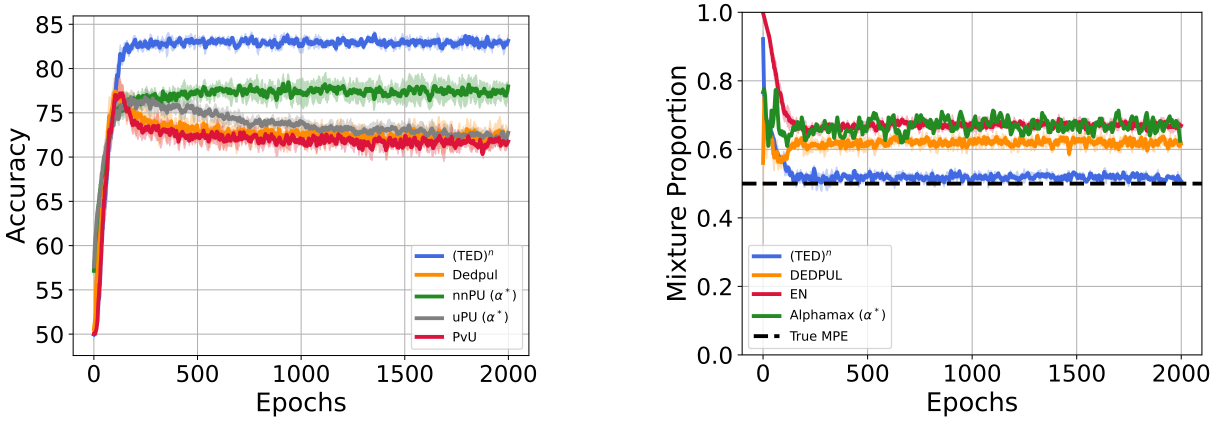


Figure 4.3: Epoch wise results with ResNet-18 trained on binary-CIFAR when α is 0.5. For both classification and MPE, (TED)^n substantially improves over existing methods. Additionally, (TED)^n maintains the superior performance till convergence removing the need for early stopping. Results aggregated over 3 seeds.

classifier (Byrd and Lipton, 2019). Therefore, we may not need importance-reweighting with (TED)^n on separable datasets. Consequently, following earlier works (Du Plessis et al., 2015; Kiryo et al., 2017) we do not re-weight the loss with our (TED)^n procedure. In future work, a simple empirical strategy can be explored where we first obtain an estimate of $\widehat{w}\alpha$ by running the full (TED)^n procedure till convergence and then discarding the (TED)^n classifier, we use estimate $\widehat{w}\alpha$ to train a fresh classifier with CVIR procedure.

Finally, we discuss an important distinction with Dedpul which is also an alternating procedure. While in our algorithm, after updating mixture proportion estimate we retrain the classifier, Dedpul fixes the classifier, obtains output probabilities and then iteratively updates the mixture proportion estimate (prior) and output probabilities (posterior). Dedpul doesn't re-train the classifier.

4.6 Experiments

Having presented our PU learning and MPE algorithms, we now compare their performance with other methods empirically. We mainly focus on vision and text datasets in our experiments. We include results on UCI datasets in App. C.7.6.

Datasets and Evaluation We simulate PU tasks on CIFAR-10 (Krizhevsky and Hinton, 2009), MNIST (LeCun et al., 1998), and IMDb sentiment analysis (Maas et al., 2011) datasets. We consider binarized versions of CIFAR-10 and MNIST. On CIFAR-10 dataset, we consider two classification problems: (i) binarized CIFAR, i.e., first 5 classes vs rest; (ii) Dog vs Cat in CIFAR. Similarly, on MNIST, we consider: (i) binarized MNIST, i.e., digits 0-4 vs 5-9; (ii) MNIST17, i.e., digit 1 vs 7. IMDb dataset is binary. For MPE, we use a held out PU validation set. To evaluate PU classifiers, we calculate accuracy on held out positive versus negative dataset. For baselines that suffer from issues due to overfitting on

Dataset	Model	$(\text{TED})^n$	BBE*	DEDPUL*	AlphaMax*	EN*	KM2	TiCE
Binarized CIFAR	ResNet	**0.026**	0.091	0.091	0.125	0.192	0.168	0.251
	All Conv	0.042	**0.037**	0.052	0.09	0.221		
	MLP	0.225	0.177	**0.138**	0.3	0.372		
CIFAR Dog vs Cat	ResNet	**0.078**	0.176	0.170	0.17	0.226	0.331	0.286
	All Conv	**0.066**	0.128	0.115	0.19	0.250		
Binarized MNIST	MLP	**0.024**	0.032	0.031	0.090	0.080	0.029	0.056
MNIST17	MLP	**0.003**	0.023	0.021	0.075	0.028	0.022	0.043
IMDb	BERT	**0.008**	0.011	0.016	0.07	0.12	-	-

Table 4.1: Absolute estimation error when α is 0.5. "*" denote oracle early stopping as defined in Sec. 4.6. $(\text{TED})^n$ significantly reduces estimation error when compared with existing methods. Results reported by aggregating absolute error over 10 epochs and 3 seeds. For aggregate numbers with standard deviation see App. C.7.5.

unlabeled data, we report results with an *oracle early stopping* criterion. In particular, we report the accuracy averaged over 10 iterations of the best performing model as evaluated on positive versus negative data. Note that we use this oracle stopping criterion only for previously proposed methods and not for methods proposed in this work. This allows us to compare $(\text{TED})^n$ with the best performance achievable by previous methods that suffer from over-fitting issues. With nnPU and $(\text{TED})^n$, we report average accuracy over 10 iterations of the final model.

Architectures For CIFAR datasets, we consider (fully connected) multilayer perceptrons (MLPs) with ReLU activations, all convolution nets (Springenberg et al., 2014), and ResNet18 (He et al., 2016). For MNIST, we consider multilayer perceptrons (MLPs) with ReLU activations For the IMDb dataset, we fine-tune an off-the-shelf uncased BERT model (Devlin et al., 2019; Wolf et al., 2020). We did not tune hyperparameters or the optimization algorithm—instead we use the same benchmarked hyperparameters and optimization algorithm for each dataset. For our method, we use cross-entropy loss. For uPU and nnPU, we use Adam (Kingma and Ba, 2014) with sigmoid loss. We provide additional details about the datasets and architectures in App. C.6.

Mixture Proportion Estimation First, we discuss results for MPE (Table 4.1). We compare our method with KM2, TiCE, DEDPUL, AlphaMax and EN. Following earlier works (Ivanov, 2019; Ramaswamy et al., 2016), we reduce datasets to 50 dimensions with PCA for KM2 and TiCE. We use existing implementation for other methods. For BBE, DEDPUL and Alphamax, we use the same PvU classifier as input. On CIFAR datasets, convolutional classifier based estimators significantly outperform KM2 and TiCE. In contrast, the performance of KM2 is comparable to DEDPUL on MNIST datasets. On all datasets, $(\text{TED})^n$ achieves lowest estimation error. With the same blackbox classifier

Dataset	Model	$(\text{TED})^n$ (unknown α)	CVIR (known α)	PvU* (known α)	DEDPUL* (unknown α)	nnPU (known α)	uPU* (known α)
Binarized CIFAR	ResNet	**82.7**	82.3	76.9	77.1	77.2	76.7
	All Conv	77.9	**78.1**	75.8	77.1	73.4	72.5
	MLP	64.2	**66.9**	61.6	62.6	63.1	64.0
CIFAR Dog vs Cat	ResNet	**75.2**	73.3	67.3	67.0	71.8	68.8
	All Conv	**73.0**	71.7	70.5	69.2	67.9	67.5
Binarized MNIST	MLP	95.6	**96.3**	94.2	94.8	96.1	95.2
MNIST17	MLP	**98.7**	**98.7**	96.9	97.7	98.4	98.4
IMDb	BERT	**87.6**	87.4	86.1	87.3	86.2	85.9

Table 4.2: Accuracy for PvN classification with PU learning. "*" denote oracle early stopping as defined in Sec. 4.6. Results reported by aggregating over 10 epochs and 3 seeds. Both CVIR (with known MPE) and $(\text{TED})^n$ (with unknown MPE) significantly improve over previous baselines with oracle early stopping and known MPE. For aggregate numbers with standard deviation see App. C.7.5.

obtained with oracle early stopping, BBE performs similar or better than best alternate(s). Since overparamterized models start memorizing unlabeled samples negatives, the quality of MPE degrades substantially as PvU training proceeds for all methods but $(\text{TED})^n$ as in Fig. 4.3.

Classification with known MPE Now, we discuss results for classification with known α. We compare our method with uPU, nnPU, DEDPUL and PvU training. Although, we solve both MPE and classification, some comparison methods do not. Ergo, we compare our classification algorithm with known MPE (Algorithm 6).

To begin, first we note that nnPU and PvU training with CVIR doesn't need early stopping. For all other methods, we report the best performance dictated by the aformentioned oracle stopping criterion. On all datasets, PvU training with CVIR leads to improved classification performance when compared with alternate approaches (Table 4.2). Moreover, as training proceeds (Fig. 4.3), the performance of DEDPUL, PvU training and uPU substantially degrade. We repeated experiments with the early stopping criterion mentioned in DEDPUL (App. C.7.2), however, their early stopping criterion is too pessimistic resulting in poor results due to under-fitting.

Classification with unknown MPE Finally, we evaluate $(\text{TED})^n$, our alternating procedure for MPE and PU learning. Across many tasks, we observe substantial improvements over existing methods. Note that these improvements often are over an oracle early stopping baselines highlighting significance of our procedure.

In App. C.7.4, we show that our procedure is not sensitive to warm start epochs W, and in many tasks with $W = 0$, we observe minor-to-no differences in the performance of $(\text{TED})^n$.

While for the experiments in this section, we used fixed $W = 100$, in the Appendix we show behavior with varying W. We also include ablations with different mixture proportions α.

4.7 Conclusion and Future Work

In this chapter, we proposed two practical algorithms, BBE (for MPE) and CVIR optimization (for PU learning). Our methods outperform others empirically and BBE's mixture proportion estimates leverage black box classifiers to produce (nearly) consistent estimates with finite sample convergence guarantees whenever we possess a classifier with a (nearly) pure top bin. Moreover, (TED)^n combines our procedures in an iterative fashion, achieving further gains. We expand our work to the multiclass problem in the next chapter, bridging work on label shift and PU learning.

Chapter 5

Domain Adaptation under Open Set Label Shift

Based on Garg et al. (2022a): Saurabh Garg, Sivaraman Balakrishnan, and Zachary Lipton. Domain adaptation under open set label shift. Advances in Neural Information Processing Systems, 2022.

Abstract

In this chapter, we extend the PU learning problem to allow multiple classes. We introduce the problem of domain adaptation under Open Set Label Shift (OSLS) where the label distribution can change arbitrarily and a new class may arrive during deployment, but the class-conditional distributions $p(x|y)$ are domain-invariant. OSLS subsumes domain adaptation under label shift and PU learning. The learner's goals here are two-fold: (a) estimate the target label distribution, including the novel class; and (b) learn a target classifier. First, we establish necessary and sufficient conditions for identifying these quantities. Second, motivated by advances in label shift and PU learning, we propose practical methods for both tasks that leverage black-box predictors. Unlike typical open set domain adaptation problems, which tend to be ill-posed and amenable only to heuristics, OSLS offers a well-posed problem amenable to more principled machinery. Experiments across numerous semi-synthetic benchmarks on vision, language, and medical datasets demonstrate that our methods consistently outperform open set domain adaptation baselines, achieving 10–25% improvements in target domain accuracy. Finally, we analyze the proposed methods, establishing finite-sample convergence to the true label marginal and convergence to optimal classifier for linear models in a Gaussian setup. Code is available at url..

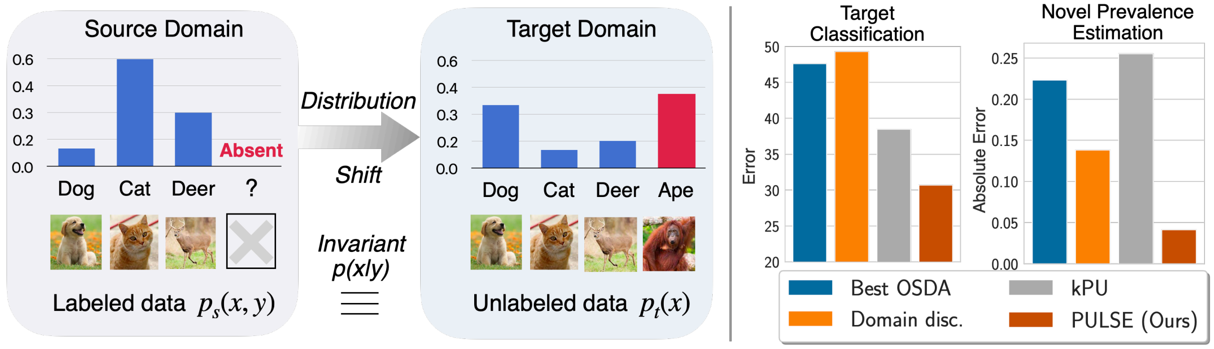


Figure 5.1: **Left:** *Domain Adaptation under OSLS*. An instantiation of OSDA that applies label shift assumption but allows for a new class to show up in target domain. **Right:** *Aggregated results across seven semi-synthetic benchmark datasets*. For both target classification and novel class prevalence estimation, PULSE significantly outperforms other methods (lower error is better). For brevity, we only include result for the best OSDA method. For detailed comparison, refer Sec. 5.7.

5.1 Introduction

Literature on Open Set Domain Adaptation (OSDA) seeks to handle cases with previously unseen classes (Baktashmotlagh et al., 2019; Cao et al., 2019b; Fu et al., 2020; Lian et al., 2019; Panareda Busto and Gall, 2017; Saito et al., 2018b; 2020; Tan et al., 2019; You et al., 2019)). Given access to labeled *source* data and unlabeled *target* data, the goal in OSDA is to adapt classifiers in general settings where previous classes can shift in prevalence (and even appearance), and novel classes separated out from those previously seen can appear. Most work on OSDA is driven by the creation of and progress on benchmark datasets (e.g., DomainNet, OfficeHome). Existing OSDA methods are heuristic in nature, addressing settings where the right answers seem intuitive but are not identified mathematically. However, absent assumptions on: (i) the nature of distribution shift among source classes and (ii) the relation between source classes and novel class, standard impossibility results for domain adaptation condemn us to guesswork (Ben-David et al., 2010c).

In this chapter, we introduce domain adaptation under Open Set Label Shift (OSLS), a coherent instantiation of OSDA that applies the label shift assumption but allows for a new class to show up in the target distribution. Formally, the label distribution may shift between source and target $p_s(y) \neq p_t(y)$, but the class-conditional distributions among previously seen classes may not (i.e., $\forall y \in \{1, 2, \ldots, k\}, p_s(x|y) = p_t(x|y)$). Moreover, a new class $y = k + 1$ may arrive in the target period. Notably, OSLS subsumes label shift (Lipton et al., 2018b; Saerens et al., 2002; Storkey, 2009) (when $p_t(y = k + 1) = 0$) and learning from Positive and Unlabeled (PU) data (De Comité et al., 1999; Elkan and Noto, 2008; Letouzey et al., 2000) (when $k = 1$). As with label shift and PU learning, our goals are two-fold. Here, we must (i) estimate the target label distribution $p_t(y)$ (including the novel class prevalence); (ii) train a $(k + 1)$-way target-domain classifier.

First, we characterize when the parameters of interest are identified (Sec. 5.4). Namely, we define a (necessary) *weak positivity* condition, which states that there exists a subset of each label's support that has zero probability mass under the novel class and that the submatrix of $p(x|y)$ consisting only of rows in that subset is full rank. Moreover, we prove that weak positivity alone is not sufficient. We introduce two sufficient conditions: *strong positivity* and *separability*, either of which (independently) ensures identifiability.

Focusing on cases with strong positivity, we show that OSLS reduces to k PU learning problems (Sec. 5.5). However, we demonstrate that straightforward applications of this idea fail because (i) bias accumulates across the k mixture proportion estimates leading to grossly underestimating the novel class's prevalence; and (ii) naive combinations of the k PU classifiers are biased and inaccurate.

Thus motivated, we propose the PULSE framework, which combines methods from Positive and Unlabeled learning and Label Shift Estimation, yielding two-stage techniques for both label marginal estimation and classification (Sec. 5.6). Our methods build on recent advances in label shift (Alexandari et al., 2021; Azizzadenesheli et al., 2019; Garg et al., 2020a; Lipton et al., 2018b) and PU learning (Garg et al., 2021b; Ivanov, 2019; Kiryo et al., 2017), that leverage appropriately chosen black-box predictors to avoid the curse of dimensionality. PULSE first estimates the label shift among previously seen classes, and then re-samples the source data to formulate a single PU learning problem between (reweighted) source and target data to estimate fraction of novel class and to learn the target classifier. In particular, our procedure builds on the BBE and CVIR techniques proposed in Garg et al. (2021b). PULSE is simple to implement and compatible with arbitrary hypothesis classes (including deep networks).

We conduct extensive semi-synthetic experiments adapting seven benchmark datasets spanning vision (CIFAR10, CIFAR100, Entity30), natural language (Newsgroups-20), biology (Tabula Muris), and medicine (DermNet, BreakHis) (Sec. 5.7). Across numerous data modalities, draws of the label distributions, and model architectures, PULSE consistently outperforms generic OSDA methods, improving by 10–25% in accuracy on target domain. Moreover, PULSE outperforms methods that naively solve k PU problems on both label distribution estimation and classification.

Finally, we analyze our framework (Sec. 5.8). First, we extend Garg et al. (2021b)'s analysis of BBE to derive finite-sample error bounds for our estimates of the label marginal. Next, we develop new analyses of the CVIR objective (Garg et al., 2021b) that PULSE relies in the classification stage. Focusing on a Gaussian setup and linear models optimized by gradient descent, we prove that CVIR converges to a true positive versus negative classifier in population. Addressing the overparameterized setting where parameters exceed dataset size, we conduct an empirical study that helps to elucidate why, on separable data, CVIR outperforms other consistent objectives, including nnPU (Kiryo et al., 2017) and uPU (Du Plessis et al., 2015).

5.2 Related Work

(Closed Set) Domain Adaptation (DA) Under DA, the goal is to adapt a predictor from a source distribution with labeled data to a target distribution from which we observe only unlabeled examples. DA is classically explored under two distribution shift scenarios (Storkey, 2009): (i) Covariate shift (Cortes and Mohri, 2014; Cortes et al., 2010; Gretton et al., 2009; Zadrozny, 2004; Zhang et al., 2013) where $p(y|x)$ remains invariant among source and target; and (ii) Label shift (Alexandari et al., 2021; Azizzadenesheli et al., 2019; Garg et al., 2020a; Lipton et al., 2018b; Saerens et al., 2002; Zhang et al., 2021) where $p(x|y)$ is shared across source and target. In these settings most theoretical analysis requires that the target distribution's support is a subset of the source support (Ben-David et al., 2010c). However, recent empirically work in DA (Ganin et al., 2016; Long et al., 2015; 2017; Sohn et al., 2020; Sun and Saenko, 2016; Sun et al., 2017; Zhang et al., 2018c; 2019) focuses on settings motivated by benchmark datasets (e.g., WILDS (Koh et al., 2021; Sagawa et al., 2021), Office-31 (Saenko et al., 2010) OfficeHome (Venkateswara et al., 2017), DomainNet (Peng et al., 2019)) where such overlap assumptions are violated. Instead, they rely on some intuitive notion of semantic equivalence across domains. These problems are not well-specified and in practice, despite careful hyperparameter tuning, these methods often do not improve over standard empirical risk minimization on source data alone for practical, and importantly, previously unseen datasets (Sagawa et al., 2021).

Open Set Domain Adaptation (OSDA) OSDA (Bendale and Boult, 2015; Panareda Busto and Gall, 2017; Scheirer et al., 2013) extends DA to settings where along with distribution shift among previously seen classes, we may observe a novel class in the target data. This setting is also known as *universal domain adaptation* (Saito et al., 2020; You et al., 2019). Rather than making precise assumptions about the nature of shift between source and target as in OSLS, the OSDA literature is primarily governed by semi-synthetic problems on benchmark DA datasets (e.g. DomainNet, Office-31 and OfficeHome). Numerous OSDA methods have been proposed (Baktashmotlagh et al., 2019; Bucci et al., 2020; Cao et al., 2019b; Fu et al., 2020; Lian et al., 2019; Saito et al., 2018b; 2020; Tan et al., 2019; You et al., 2019). At a high level, most OSDA methods perform two steps: (i) align source and target representation for previously seen classes; and (ii) train a discrimination to reject novel class from previously seen classes. The second step typically uses novelty detection heuristics to identify novel samples.

Other related work A separate line of work looks at the problem of Out-Of-Distribution (OOD) detection (Geifman and El-Yaniv, 2017; Hendrycks and Gimpel, 2017; Jiang et al., 2018; Lakshminarayanan et al., 2016; Ovadia et al., 2019; Zhang et al., 2020). Here, the goal is to identify novel examples, i.e., samples that lie out of the support of training distribution. The main different between OOD detection and OSDA is that in OOD detection we do not have access to unlabeled data containing a novel class. Recently, Cao et al. (2022) proposed open-world semi-supervised learning, where the task is to not only identify novel classes in target but also to separate out different novel classes in an unsupervised manner.

Our work takes a step back from the hopelessly general OSDA setup, introducing OSLS, a well-posed OSDA setting where the sought-after parameters can be identified.

5.3 Open Set Label Shift

Notation For a vector $v \in \mathbb{R}^d$, we use v_j to denote its j^{th} entry, and for an event E, we let $\mathbb{I}[E]$ denote the binary indicator of the event. By $|A|$, we denote the cardinality of set A.

Let $\mathcal{X} \in \mathbb{R}^d$ be the input space and $\mathcal{Y} = \{1, 2, \dots, k+1\}$ be the output space for multiclass classification. Let P_s and P_t be the source and target distributions and let p_s and p_t denote the corresponding probability density (or mass) functions. By $\mathbb{E}_s$ and $\mathbb{E}_t$, we denote expectations over the source and target distributions. We assume that we are given a loss function $\ell : \Delta^k \times \mathcal{Y} \to \mathbb{R}$, such that $\ell(z, y)$ is the loss incurred by predicting z when the true label is y. Unless specified otherwise, we assume that ℓ is the cross entropy loss. As in standard unsupervised domain adaptation, we are given independently and identically distributed (iid) samples from labeled source data $\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\} \sim \mathrm{P}_s^n$ and iid samples from unlabeled target data $\{x_{n+1}, x_{n+2}, \dots, x_{n+m}\} \sim \mathrm{P}_t^m$.

Before formally introducing OSLS, we describe label shift and PU learning settings. Under label shift, we observe data from k classes in both source and target where the conditional distribution remain invariant (i.e., $p_s(x|y) = p_t(x|y)$ for all classes $y \in [1, k]$) but the target label marginal may change (i.e., $p_t(y) \neq p_s(y)$). Additionally, for all classes in source have a non-zero support , i.e., for all $y \in [1, k]$, $p_s(y) \geqslant c$, where $c > 0$. Under PU learning, we possess labeled source data from a positive class and unlabeled target data from a mixture of positive and negative class with a goal of learning a positive-versus-negative classifier on target. We now introduce the OSLS setting:

Definition 5.3.1 (Open set label shift). *Define $\mathcal{Y}_t = \mathcal{Y}$ and $\mathcal{Y}_s = \mathcal{Y} \backslash \{k+1\}$. Under OSLS, the label distribution among source classes $\mathcal{Y}_s$ may change but the class conditional $p(x|y)$ for those classes remain invariant between source and target, and the target domain may contain a novel class, i.e.,*

$$p_s(x|y = j) = p_t(x|y = j) \quad \forall j \in \mathcal{Y}_s \qquad \textit{and} \qquad p_s(y = k+1) = 0\,. \tag{5.1}$$

Additionally, we have non-zero support for all k (previously-seen) labels in the source distribution, i.e., for all $y \in \mathcal{Y}_s$, $p_s(y) \geqslant c$ for some $c > 0$.

Note that the label shift and PU learning problems can be obtained as special cases of OSLS. When no novel class is observed in target (i.e., when $p_t(y = k+1) = 0$), we recover the label shift problem, and when we observe only one class in source (i.e., when $k = 1$), the OSLS problem reduces to PU learning. Under OSLS, our goal naturally breaks down into two tasks: (i) estimate the target label marginal $p_t(y)$ for each class $y \in \mathcal{Y}$; (ii) train a classifier $f : \mathcal{X} \to \Delta^k$ to approximate $p_t(y|x)$.

5.4 Identifiablity of OSLS

We now introduce conditions for OSLS, under which the solution is identifiable. Throughout the section, we will assume access to population distribution for labeled source data and unlabeled target data, i.e., $p_s(x, y)$ and $p_t(x)$ is given. To keep the discussion simple, we assume finite input domain $\mathcal{X}$ which can then be relaxed to continuous inputs. We relegate proofs to App. D.2.

We first make a connection between target label marginal $p_t(y)$ estimation and learning the target classifier $p_t(y|x)$ showing that recovering $p_t(y)$ is enough to identify $p_t(y|x)$. In population, given access to $p_t(y)$, the class conditional $p_t(x|y = k + 1)$ can be obtained in closed form as $\left(p_t(x) - \sum_{j=1}^{k} p_t(y=j) p_s(x|y=j)\right) / p_t(y=k+1)$. We can then apply Bayes rule to obtain $p_t(y|x)$. Henceforth, we will focus our discussion on identifiability of $p_t(y)$ which implies identifiability of $p_t(y|x)$. In following proposition, we present *weak positivity*, a necessary condition for $p_t(y)$ to be identifiable. [Necessary conditions] Assume $p_t(y) > 0$ for all $y \in \mathcal{Y}_t$. Then $p_t(y)$ is identified only if $p_t(x|y = k + 1)$ and $p_s(x|y)$ for all $y \in \mathcal{Y}_s$ satisfy weak positivity, i.e., there must exists a subdomain $X_{\text{wp}} \subset X$ such that:

(i) $p_t(X_{\text{wp}}|y = k + 1) = 0$; and

(ii) the matrix $[p_s(x|y)]_{x \in X_{\text{wp}}, y \in \mathcal{Y}_s}$ is full column-rank.

Intuitively, Proposition 5.4 states that if the target marginal doesn't lie on the vertex of the simplex Δ^k, then their must exist a subdomain X_{wp} where the support of novel class is zero and within X_{wp}, $p_t(y)$ for source classes is identifiable. While it may seem that existence of a subdomain X_{wp} is enough, we show that for the OSLS problem, existence doesn't imply uniqueness. In App. D.2.1, we construct an example, where the weak positivity condition is not sufficient. In that example, we show that there can exist two subdomains X_{wp} and X'_{wp} satisfying weak positivity, both of which lead to separate solutions for $p_t(y)$. Next, we extend weak positivity to two stronger conditions, either of which (alone) implies identifiability. [Sufficient conditions] The target marginal $p_t(y)$ is identified if for all $y \in \mathcal{Y} \backslash \{k + 1\}$, $p_t(x|y = k + 1)$ and $p_s(x|y)$ satisfy either:

(i) Strong positivity, i.e., there exists $X_{\text{sp}} \subset \mathcal{X}$ such that $p_t(X_{\text{sp}}|y = k + 1) = 0$ and the matrix $[p_s(x|y)]_{x \in X_{\text{sp}}, y \in \mathcal{Y}_s}$ is full-rank and diagonal; or

(ii) Separability, i.e., there exists $X_{\text{sep}} \subset \mathcal{X}$, such that $p_t(X_{\text{sep}}|y = k + 1) = 0$, $p_s(X_{\text{sep}}) = 1$, and the matrix $[p_s(x|y)]_{x \in X_{\text{sep}}, y \in \mathcal{Y}_s}$ is full column-rank.

Strong positivity generalizes the irreducibility condition (Blanchard et al., 2010), which is sufficient for identifiability under PU learning, to k PU learning problems. Note that while the two conditions in Proposition 5.4 overlap, they cover independent set of OSLS problems. Informally, strong positivity extends weak positivity by making an additional assumption that the matrix formed by $p(x|y)$ on inputs in X_{wp} is diagonal and the separability assumption extends the weak positivity condition to the full input domain of source classes instead of just X_{wp}. Both of these conditions identify a support region of $\mathcal{X}$ which purely belongs to source classes where we can either individually estimate the proportion of each source classes (i.e., under strong positivity) or jointly estimate the proportion (i.e., under separability).

To extend our identifiability conditions for continuous distributions, the linear independence conditions on the matrix $[p_s(x|y)]_{x\in X_{\text{sep}}, y\in\mathcal{Y}_s}$ has the undesirable property of being sensitive to changes on sets of measure zero. We may introduce stronger notions of linear independence as in Lemma 1 of Garg et al. (2020a). We discuss this in App. D.2.2.

5.5 Reduction of OSLS to k PU Problems

Under the strong positivity condition, the OSLS problem can be broken down into k PU problems as follows: By treating a given source class $y_j \in \mathcal{Y}_s$ as *positive* and grouping all other classes together as *negative* we observe that the unlabeled target data is then a mixture of data from the positive and negative classes. This yields a PU learning problem and the corresponding mixture proportion is the fraction $p_t(y=j)$ (proportion of class y_j) among the target data. By iterating this process for all source classes, we can solve for the entire target label marginal $p_t(y)$. Thus, OSLS reduces to k instances of PU learning problem. Formally, note that $p_t(x)$ can be written as:

$$p_t(x) = p_t(y=j)p_s(x|y=j) + (1-p_t(y=j))\left(\sum\nolimits_{i\in\mathcal{Y}\setminus\{j\}} \frac{p_t(y=i)}{1-p_t(y=j)} p_s(x|y=i)\right), \tag{5.2}$$

individually for all $j \in \mathcal{Y}_s$. By repeating this reduction for all classes, we obtain k separate PU learning problems. Hence, a natural choice is to leverage this structure and solve k PU problems to solve the original OSLS problem. In particular, for each class $j \in \mathcal{Y}_s$, we can first estimate its prevalence $\widehat{p}_t(y=j)$ in the unlabeled target. Then the target marginal for the novel class is given by $\widehat{p}_t(y=k+1) = 1 - \sum_{i=1}^{k} \widehat{p}_t(y=i)$. Similarly, for classification, we can train k PU learning classifiers f_i, where f_i is trained to classify a source class i versus others in target. An example is classified as belonging to the class $y = k+1$, if it rejected by all classifiers f_i as other in target. We explain this procedure more formally in App. D.1.

This reduction has been mentioned in past work (Sanderson and Scott, 2014; Xu et al., 2017). However, to the best of our knowledge, no previous work has empirically investigated both classification and target label marginal estimation jointly. Sanderson and Scott (2014) focuses only on target marginal estimation for tabular datasets and Xu et al. (2017) assumes that the target marginal is known and only trains k separate PU classifiers.

In our work, we perform the first large scale experiments to evaluate efficacy of the reduction of the OSLS problem to k-PU problems. With plugin state-of-the-art PU learning algorithms, we observe that this naive reduction doesn't scale to datasets with large number of classes because of error accumulation in each of the k MPEs and k one-versus-other PU classifiers. To mitigate the error accumulation problem, we propose the PULSE framework in the next section.

5.6 The PULSE Framework for OSLS

We begin with presenting our framework for OSLS problem under strong positivity condition. First, we explain the structure of OSLS that we leverage in PULSE framework and then elaborate design decisions we make to exploit the identified structure.

Overview of PULSE framework Rather than simply dividing each OSLS instance into k PU problems, we exploit the joint structure of the problem to obtain a *single* PU learning problem. To begin, we note that if only we could apply a *label shift correction* to source, i.e., re-sample source classes according to their relative proportion in the target data, then we could subsequently consider the unlabeled target data as a mixture of (i) the (reweighted) source distribution; and (ii) the novel class distribution (i.e., $p_t(x|y = k + 1)$). Formally, we have

$$\begin{aligned} p_t(x) = \sum_{j \in \mathcal{Y}_t} p_t(y = j) p_t(x|y = j) &= \sum_{j \in \mathcal{Y}_s} \frac{p_t(y = j)}{p_s(y = j)} p_s(x, y = j) + p_t(x|y = k + 1) p_t(y = k + 1) \\ &= (1 - p_t(y = k + 1)) p'_s(x) + p_t(y = k + 1) p_t(x|y = k + 1) \, , \end{aligned} \tag{5.3}$$

where $p'_s(x)$ is the label-shift-corrected source distribution, i.e., $p'_s(x) = \sum_{j \in \mathcal{Y}_s} w(j) p_s(x, y = j)$, where $w(j) = ({p_t(y=j)}/{\sum_k p_t(y=k)})/p_s(y = j)$ for all $j \in \mathcal{Y}_s$. Intuitively, $p'_t(j) = {p_t(y=j)}/{\sum_k p_t(y=k)}$ is re-normalized label distribution in target among source classes and $w(j)$'s are the importance weights. Hence, after applying a label shift correction to the source distribution $p'_s(x)$, we have reduced the OSLS problem to a *single* PU learning problem, where $p'_s(x)$ plays the part of the positive distribution and $p_t(x|y = k + 1)$ acts as negative distribution with mixture coefficients $1 - p_t(y = k + 1)$ and $p_t(y = k + 1)$ respectively. We now discuss our methods (i) to estimate the importance ratios $w(y)$; and (ii) to tackle the PU learning instance obtained from OSLS.

Label shift correction: Target marginal estimation among source classes While traditional methods for estimating label shift breakdown in high dimensional settings (Zhang et al., 2013), recent methods exploit black-box classifiers to avoid the curse of dimensionality (Alexandari et al., 2021; Azizzadenesheli et al., 2019; Lipton et al., 2018b). However, these recent techniques require overlapping label distributions, and a direct application would require demarcation of samples from $p'_s(x)$ sub-population in target, creating a cyclic dependency. Instead, to estimate the relative proportion of previously seen classes in target, we leverage the k PU reduction described in Sec. 5.5 with two crucial distinctions. First, we normalize the obtained estimates of fraction previously seen classes to obtain the relative proportions in $p'_s(y)$. In particular, we do not leverage the estimates of previously seen class proportions in target to directly estimate the proportion of novel class which avoids issues due to error accumulation. Second, we exploit a k-way source classifier f_s trained on labeled source data instead of training k one-versus-other PU classifiers. We tailor the recently proposed Best Bin Estimation (BBE) technique from Garg et al. (2021b). We describe the modified BBE procedure in App. D.3 (Algorithm 15). After estimating the relative fraction of source classes in target (i.e., $\widehat{w}p'_t(j) = {\widehat{w}p_t(y=j)}/{\sum_{k \in \mathcal{Y}_s} \widehat{w}p_t(y=k)}$ for all $j \in \mathcal{Y}_s$), we re-sample the source data according to $\widehat{w}p'_t(y)$ to mimic samples from distribution $p'_s(x)$.

Algorithm 8 Positive and Unlabeled learning post Label Shift Estimation (PULSE) framework

```
input : Labeled source data $\{\mathbf{X}^S, \mathbf{y}^S\}$ and unlabeled target samples $\mathbf{X}^T$.
Randomly split data into training $\{\mathbf{X}_1^S, \mathbf{y}_1^S\}$, $\mathbf{X}_1^T$ and hold out partition $\{\mathbf{X}_2^S, \mathbf{y}_2^S\}$, $\mathbf{X}_2^T$.
Train a source classifier $f_s$ on labeled source data $\{\mathbf{X}_1^S, \mathbf{y}_1^S\}$.
Estimate label shift $\widehat{w}p_t'(y = j) = \frac{\widehat{w}p_t(y=j)}{\sum_{k \in \mathcal{Y}_s} \widehat{w}p_t(y=k)}$ using Algorithm 15 and hence importance ratios $\widehat{w}w(j)$ among source classes $j \in \mathcal{Y}_s$.
Re-sample training source data according to label distribution $\widehat{w}p_t'$ to get $\{\widetilde{\mathbf{X}}_1^S, \widetilde{\mathbf{y}}_1^S\}$ and $\{\widetilde{\mathbf{X}}_2^S, \widetilde{\mathbf{y}}_2^S\}$.
Using Algorithm 16, train a discriminator $f_d$ and estimate novel class fraction $\widehat{w}p_t(y = k+1)$.
Assign $[f_t(x)]_j = (f_d(x)) \frac{\widehat{w}w(j) \cdot [f_s(x)]_j}{\sum_{k \in \mathcal{Y}_s} \widehat{w}w(k) \cdot [f_s(x)]_k}$ for all $j \in \mathcal{Y}_s$ and $[f_t(x)]_{k+1} = 1 - f_d(x)$.
    And for all $j \in \mathcal{Y}_s$, assign $\widehat{w}p_t(y = j) = (1 - \widehat{w}p_t(y = k+1)) \cdot \widehat{w}p_t'(y = j)$.
output : Target marginal estimate $\widehat{w}p_t \in \Delta^k$ and target classifier $f_t(\cdot) \in \Delta^k$.
```

PU Learning: Separating the novel class from previously seen classes After obtaining a PU learning problem instance, we resort to PU learning techniques to (i) estimate the fraction of novel class $p_t(y = k+1)$; and (ii) learn a binary classifier $f_d(x)$ to discriminate between label shift corrected source $p_s'(x)$ and novel class $p_t(x|y = k+1)$. With traditional methods for PU learning involving domain discrimination, over-parameterized models can memorize the positive instances in unlabeled, assigning them confidently to the negative class, which can severely hurt generalization on PN data (Garg et al., 2021b; Kiryo et al., 2017). Rather, we employ Conditional Value Ignoring Risk (CVIR) loss proposed in Garg et al. (2021b) which was shown to outperform alternative approaches. First, we estimate the proportion of novel class $\widehat{w}p_t(y = k+1)$ with BBE. Next, given an estimate $\widehat{w}p_t(y = k+1)$, CVIR objective discards the highest loss $(1 - \widehat{w}p_t(y = k+1))$ fraction of examples on each training epoch, removing the incentive to overfit to the examples from $p_s'(x)$. Consequently, we employ the iterative procedure that alternates between estimating the prevalence of novel class $\widehat{w}p_t(y = k+1)$ (with BBE) and minimizing the CVIR loss with estimated fraction of novel class. We detail this procedure in App. D.3 (Algorithm 16).

Combining PU learning and label shift correction Finally, to obtain a $(k+1)$-way classifier $f_t(x)$ on target we combine discriminator f_d and source classifier f_s with importance-reweighted label shift correction. In particular, for all $j \in \mathcal{Y}_s$, $[f_t(x)]_j = (f_d(x)) \frac{w(j) \cdot [f_s(x)]_j}{\sum_{k \in \mathcal{Y}_s} w(k) \cdot [f_s(x)]_k}$ and $[f_t(x)]_{k+1} = 1 - f_d(x)$. Overall, our approach outlined in Algorithm 8 proceeds as follows: First, we estimate the label shift among previously seen classes. Then we employ importance re-weighting of source data to formulate a single PU learning problem to estimate the fraction of novel class $\widehat{w}p_t(y = k+1)$ and to learn a discriminator f_d for the novel class. Combining discriminator and label shift corrected source classifier we get $(k+1)$-way target classifier. We analyse crucial steps in PULSE in

Sec. 5.8.

Our ideas for PULSE framework can be extended to separability condition since (5.3) continues to hold. However, in our initial experiments, we observe that techniques proposed under strong positivity were empirically stable and outperform methods developed under separability. This is intuitive for many benchmark datasets where it is natural to assume that for each class there exists a subdomain that only belongs to that class. We describe this in more detail in App. D.3.1.

5.7 Experiments

Baselines We compare PULSE with several popular methods from OSDA literature. While these methods are not specifically proposed for OSLS, they are introduced for the more general OSDA problem. In particular, we make comparions with DANCE (Saito et al., 2020), UAN (You et al., 2019), CMU (Fu et al., 2020), STA (Liu et al., 2019a), Backprop-ODA (or BODA) (Saito et al., 2018b). We use the open source implementation available at `https://github.com/thuml`. For alternative baselines, we experiment with source classifier directly deployed on the target data which may contain novel class and label shift among source classes (referred to as *source-only*). We also train a domain discriminator classifier for source versus target (referred to as *domain disc.*). This is adaptation of PU learning baseline(Elkan and Noto, 2008) which assumes no label shift among source classes. Finally, per the reduction presented in Sec. 5.5, we train k PU classifiers (referred to as *k-PU*). We include detailed description of each method in App. D.6.1.

Datasets We conduct experiments with seven benchmark classification datasets across vision, natural language, biology and medicine. For each dataset, we simulate an OSLS problem as described in next paragraph. For vision, we use CIFAR10, CIFAR100 (Krizhevsky and Hinton, 2009) and Entity30 (Santurkar et al., 2021). For language, we experiment with Newsgroups-20 (`http://qwone.com/~jason/20Newsgroups/`) dataset. Additionally, inspired by applications of OSLS in biology and medicine, we experiment with Tabula Muris (Consortium et al., 2020) (Gene Ontology prediction), Dermnet (skin disease prediction `https://dermnetnz.org/`), and BreakHis (Spanhol et al., 2015) (tumor cell classification). These datasets span language, image and table modalities. We provide interpretation of OSLS problem for each dataset along with other details in App. D.6.2.

OSLS Setup To simulate an OSLS problem, we experiment with different fraction of novel class prevalence, source label distribution, and target label distribution. We randomly choose classes that constitute the novel target class. After randomly choosing source and novel classes, we first split the training data from each source class randomly into two partitions. This creates a random label distribution for shared classes among source and target. We then club novel classes to assign them a new class (i.e. $k + 1$). Finally, we throw away labels for the target data to obtain an unsupervised DA problem. We repeat the same process on iid hold out data to obtain validation data with no target labels.

Training and Evaluation We use Resnet18 (He et al., 2016) for CIFAR10, CIFAR100, and Entity30. For newsgroups, we use a convolutional architecture. For Tabular Muris and MNIST, we use a fully connected MLP. For Dermnet and BreakHis, we use Resnet-50. For all methods, we use the same backbone for discriminator and source classifier. For kPU, we use a separate final layer for each class with the same backbone. We use default hyperparameters for all methods. For OSDA methods, we use default method specific hyperparameters introduced in their works. Since OSDA methods do not estimate the prevalence of novel class explicitly, we use the fraction of examples predicted in class $k + 1$ as a surrogate. We train models till the performance on validation source data (labeled) ceases to increase. Unlike OSDA methods, note that we do not use early stopping based on performance on held-out labeled target data. To evaluate classification performance, we report target accuracy on all classes, seen classes and the novel class. For novel class prevalence estimation, we report absolute difference between the true and estimated marginal. We open-source our code and by simply changing a single config file, new OSLS setups can be generated and experimented with. We provide precise details about hyperparameters, OSLS setup for each dataset and code in App. D.6.3.

Results Across different datasets, we observe that PULSE consistently outperforms other methods for the target classification and novel prevalence estimation (Table 5.1). For detection of novel classes (Acc (Novel) column), kPU achieves superior performance as compared to alternative approaches because of its bias to default to $(k + 1)^{\text{th}}$ class. This is evident by the sharp decrease in performance on previously seen classes. For each dataset, we plot evolution of performance with training in App. D.6.4. We observe more stability in performance of PULSE as compared to other methods.

We observe that with default hyperparameters, popular OSDA methods significantly under perform as compared to PULSE. We hypothesize that the primary reasons underlying the poor performance of OSDA methods are (i) the heuristics employed to detect novel classes; and (ii) loss functions incorporated to improve alignment between examples from common classes in source and target. To detect novel classes, a standard heuristic employed in popular OSDA methods involves thresholding uncertainty estimates (e.g., prediction entropy, softmax confidence (Fu et al., 2020; Saito et al., 2020; You et al., 2019)) at a predefined threshold κ. However, a fixed κ, may not for different datasets and different fractions of the novel class. In App. D.6.5, we ablate by (i) removing loss function terms incorporated with an aim to improve source target alignment; and (ii) vary threshold κ and show improvements in performance of these methods. In contrast, our two-stage method PULSE, first estimates the fraction of novel class which then guides the classification of novel class versus previously seen classes avoiding the need to guess κ.

Ablations Different datasets, in our setup span different fraction of novel class prevalence ranging from 0.22 (in CIFAR10) to 0.64 (in Tabula Muris). For each dataset, we perform more ablations on the novel class proportion in App. D.6.6. For kPU and PULSE, in the main paper, we include results with BBE and CVIR (Garg et al., 2021b). In App. D.6.8, we perform experiments with alternative PU learning approaches and highlight the superiority of BBE and CVIR over other methods. Moreover, since we have access to unlabeled target data, we experiment with SimCLR (Chen et al., 2020a) pre-training on the mixture of

Table 5.1: *Comparison of PULSE with other methods.* Across all datasets, PULSE outperforms alternatives for both target classification and novel class prevalence estimation. Acc (All) is target accuracy, Acc (Seen) is target accuracy on examples from previously seen classes, and Acc (Novel) is recall for novel examples. MPE (Novel) is absolute error for novel prevalence estimation. Results reported by averaging across 3 seeds. Detailed results for each dataset with all methods in App. D.6.4.

	CIFAR-10				**CIFAR-100**			
Method	Acc (All)	Acc (Seen)	Acc (Novel)	MPE (Novel)	Acc (All)	Acc (Seen)	Acc (Novel)	MPE (Novel)
Source-Only	67.1	87.0	-	-	46.6	66.4	-	-
UAN (You et al., 2019)	15.4	19.7	25.2	0.214	18.1	40.6	14.8	0.133
BODA (Saito et al., 2018b)	63.1	66.2	42.0	0.162	36.1	17.7	81.6	0.41
DANCE (Saito et al., 2020)	70.4	85.5	14.5	0.174	47.3	66.4	1.2	0.28
STA (Liu et al., 2019a)	57.9	69.6	14.9	0.124	42.6	48.5	34.8	0.14
CMU (Fu et al., 2020)	62.1	77.9	41.2	0.183	35.4	46.0	15.5	0.161
Domain Disc. (Elkan and Noto, 2008)	47.4	87.0	30.6	0.331	45.8	66.5	39.1	**0.046**
k-PU	83.6	79.4	**98.9**	0.036	36.3	22.6	**99.1**	0.298
PULSE (Ours)	**86.1**	**91.8**	88.4	**0.008**	**63.4**	**67.2**	63.5	0.078

	Entity30		**Newsgroups20**		**Tabula Muris**		**BreakHis**		**DermNet**	
Method	Acc (All)	MPE (Novel)	Acc (All)	MPE (Novel)	Acc (All)	MPE (Novel)	Acc (All)	MPE (Novel)	Acc (All)	MPE (Novel)
Source-Only	32.0	-	39.3	-	33.8	-	70.0	-	41.4	-
BODA (Saito et al., 2018b)	42.2	0.189	43.4	0.16	76.5	0.079	71.5	0.077	43.8	0.207
Domain Disc.	43.2	0.135	50.9	0.176	73.0	0.071	56.5	0.091	40.6	0.083
k-PU	50.7	0.394	52.1	0.373	85.9	0.307	75.6	**0.059**	46.0	0.313
PULSE (Ours)	**58.0**	**0.054**	**62.2**	**0.061**	**87.8**	**0.058**	**79.1**	**0.054**	**48.9**	**0.043**

unlabeled source and target dataset. We include setup details and results in App. D.6.7. While pre-trained backbone architecture improves performance for all methods, PULSE continues to dominate other methods.

5.8 Analysis of PULSE Framework

In this section, we analyse key steps of our PULSE procedure for target label marginal estimation (Step 3, 5 Algorithm 8) and learning the domain discriminator classifier (Step 5, Algorithm 8). Due to space constraints, we present informal results here and relegate formal statements and proofs to App. D.4.

Theoretical analysis for target marginal estimation Building on BBE results from Garg et al. (2021b), we present finite sample results for target label marginal estimation.

When the data satisfies strong positivity, we observe that source classifiers often exhibit a threshold c_y on softmax output of each class $y \in \mathcal{Y}_s$ above which the *top bin* (i.e., $[c_y, 1]$) contains mostly examples from that class y. We give empirical evidence to this claim in App. D.4.1. Then, we show that the existence of (nearly) pure top bin for each class in f_s is sufficient for Step 3 in Algorithm 8 to produce (nearly) consistent estimates:

Theorem 5.8.1 (Informal)**.** *Assume that for each class $y \in \mathcal{Y}_s$, there exists a threshold c_y such that for the classifier f_s, if $[f_s(x)]_y > c_y$ for any x then the true label for that sample x is y. Then, we have $\|\widehat{w}p_t - p_t\| 1 \leqslant \mathcal{O}\left(\sqrt{k^3 \log(4k/\delta)/n} + \sqrt{k^2 \log(4k/\delta)/m}\right)$.*

The proof technique simply builds on the proof of Theorem 1 in Garg et al. (2021b). By assuming that we recover close to ground truth label marginal for source classes, we can also extend the above analysis to Step 5 of Algorithm 8 to show convergence of estimate $\widehat{w}p_t(y = k+1)$ to true prevalence $p_t(y = k+1)$. We discuss this further in App. D.4.3.

Theoretical analysis of CVIR in population While the CVIR loss was proposed in Garg et al. (2021b), no analysis was provided for convergence of the iterative gradient descent procedure. In our work, we show that in population on a separable Gaussian dataset, CVIR will recover the optimal classifier.

We consider a binary classification problem where we have access to positive distribution (i.e., p_p), unlabeled distribution (i.e., $p_u := \alpha p_p + (1-\alpha) p_n$), and mixture coefficient α. Making a parallel connection to Step 5 of PULSE, positive distribution p_p here refers to the label shift corrected source distribution p_s' and p_u refers to $p_t = p_t(y = k+1) p_t(x|y = k+1) + (1 - p_t(y = k+1)) p_s'(x)$. Our goal is to recover the classifier that discriminates p_p versus p_n (parallel p_s' versus $p_t(\cdot|y = k+1)$).

First we introduce some notation. For a classifier f and loss function ℓ (i.e., logistic loss), define $\text{VIR}_\alpha(f) = \inf\{\tau \in \mathbb{R} : \mathrm{P}_{x \sim p_u}(\ell(x, -1; f) \leqslant \tau) \geqslant 1 - \alpha\}$. Intuitively, $\text{VIR}_\alpha(f)$ identifies a threshold τ to capture bottom $1-\alpha$ fraction of the loss $\ell(x, -1)$ for points x sampled from p_u. Additionally, define CVIR loss as $\mathcal{L}(f, w) = \alpha \mathbb{E}_{p_p}[\ell(x, 1; f)] + \mathbb{E}_{p_u}[w(x) \ell(x, -1; f)]$ for classifier f and some weights $w(x) \in \{0, 1\}$. Formally, given a classifier f_t at an iterate t, CVIR procedure proceeds as follows:

$$w_t(x) = \mathbb{I}\left[\ell(x, -1; f_t) \leqslant \text{VIR}_\alpha(f_t)\right] , \tag{5.4}$$

$$f_{t+1} = f_t - \eta \nabla \mathcal{L}_f(f_t, w_t) . \tag{5.5}$$

We assume that x are drawn from two half multivariate Gaussian with mean zero and identity covariance, i.e., $x \sim p_p \Leftrightarrow x = \gamma_0 \theta_{\text{opt}} + z | \, \theta_{\text{opt}}^T z \geqslant 0$, and $x \sim p_n \Leftrightarrow x = -\gamma_0 \theta_{\text{opt}} + z | \, \theta_{\text{opt}}^T z < 0$, where $z \sim \mathcal{N}(0, I_d)$. Here γ_0 is the margin and $\theta_{\text{opt}} \in \mathbb{R}^d$ is the true separator. Here, we have access to distribution p_p, $p_u = \alpha p_p + (1-\alpha) p_n$, and the true proportion α.

Theorem 5.8.2 (Informal)**.** *In the data setup detailed above, a linear classifier $f(x; \theta) = \sigma\left(\theta^T x\right)$ trained with CVIR procedure as in (5.4)-(5.5) will converge to an optimal positive versus negative classifier.*

The proof uses a key idea that for any classifier θ not separating positive and negative data perfectly, the gradient in (5.5) is non-zero. Hence, convergence of the CVIR procedure (implied by smoothness of CVIR loss) implies converge to an optimal classifier. For separable datasets in general, we can extend the above analysis with some modifications to the CVIR procedure. We discuss this in App. D.4.4.

Empirical investigation in overparameterized models As noted in our ablation experiments and in Garg et al. (2021b), domain discriminator trained with CVIR outperforms classifiers trained with other consistent objectives (nnPU (Kiryo et al., 2017) and uPU (Du Plessis et al., 2015)). While the above analysis highlights consistency of CVIR procedure in population, it doesn't capture the observed empirical efficacy of CVIR over alternative methods in overparameterized models. In the Gaussian setup described above, we train overparameterized linear models to compare CVIR with other methods. We discuss precise experiments and results in App. D.5, but highlight the key takeaway here. First, we observe that when a classifier is trained to distinguish positive and unlabeled data, *early learning* happens (Arora et al., 2019a; Garg et al., 2021a; Liu et al., 2020), i.e., during the initial phase of learning classifier learns to classify positives in unlabeled correctly as positives. Next, we show that post early learning rejection of large fraction of positives from unlabeled training in equation (5.4) crucially helps CVIR.

5.9 Conclusion

In this chapter, we introduce OSLS a well-posed instantiation of OSDA that subsumes label shift and PU learning into a framework for learning adaptive classifiers. We presented identifiability conditions for OSLS and proposed PULSE, a simple and effective approach to tackle the OSLS problem. Moreover, our extensive experiments demonstrate efficacy of PULSE over popular OSDA alternatives when the OSLS assumptions are met. We highlight the brittle nature of benchmark driven progress in OSDA and hope that our work can help to stimulate more solid foundations and enable systematic progress in this area.

Chapter 6

Complementary Benefits of Contrastive Learning and Self-Training Under Distribution Shift

Based on Garg et al. (2023b): Saurabh Garg*, Amrith Setlur*, Zachary Lipton, Sivaraman Balakrishnan, Virginia Smith, and Aditi Raghunathan. Complementary Benefits of Contrastive Learning and Self-Training Under Distribution Shift. Advances in Neural Information Processing Systems, 2023.

Abstract

In the previous chapters, we focused primarily on settings where $p(x|y)$ remained invariant. In this chapter, we explore the alternate setup where covariate distribution shifts but $p(y)$ doesn't change. In these settings, self-training and contrastive learning have emerged as leading techniques for incorporating unlabeled data. However, despite the popularity and compatibility of these techniques, their efficacy in combination remains unexplored. We undertake a systematic empirical investigation of this combination, finding that (i) in domain adaptation settings, self-training and contrastive learning offer significant complementary gains; and (ii) in semi-supervised learning settings, surprisingly, the benefits are not synergistic. Across eight distribution shift datasets (*e.g.*, BREEDs, WILDS), we demonstrate that the combined method obtains 3–8% higher accuracy than either approach independently. We then theoretically analyze these techniques in a simplified model of distribution shift, demonstrating scenarios under which the features produced by contrastive learning can yield a good initialization for self-training to further amplify gains and achieve optimal performance, even when either method alone would fail.

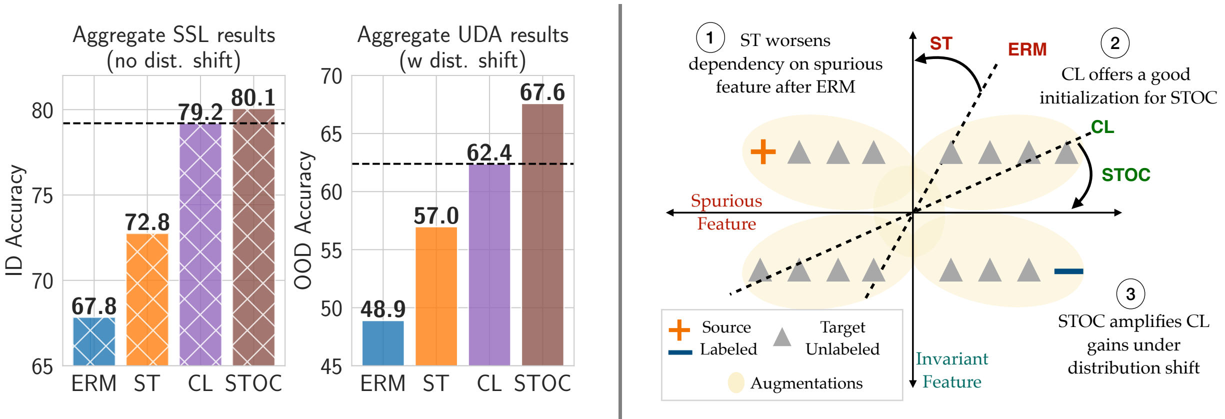
6.1 Introduction

Until now, we have focused on domain adaptation scenarios where $p(x|y)$ remained invariant. In this chapter, we will study setting where input distributions can shift due to natural perturbations in inputs. These types of perturbations include distribution shifts which simply do not follow the label shift assumption.

To address UDA in practice where inputs distribution shift, two popular methods have emerged: self-training and contrastive pretraining. Self-training (Lee et al., 2013; Scudder, 1965; Sohn et al., 2020; Wang et al., 2021a; Xie et al., 2020b) and contrastive pretraining (Caron et al., 2020; Chen et al., 2020a; Zbontar et al., 2021) were both proposed, initially, for traditional Semi-Supervised Learning (SSL) problems, where the labeled and unlabeled data are drawn from the same distribution. Here, the central challenge is statistical: to exploit the unlabeled data to learn a better predictor than one would get by training on the (small) labeled data alone. More recently, these methods have emerged as favored empirical approaches for UDA, demonstrating efficacy on many popular benchmarks (Cai et al., 2021a; Garg et al., 2023a; Sagawa et al., 2021; Shen et al., 2022). In self-training, one first learns a predictor using source labeled data. The predictor then produces pseudolabels for the unlabeled target data, and a new predictor is trained on the pseudolabeled data. Contrastive pretraining learns representations from unlabeled data by enforcing invariance to specified augmentations. These representations are subsequently used to learn a classifier. In UDA, the representations are trained on the union of the source and target data. Despite the strong performance of self-training and constrastive pretraining independently, there has been surprisingly little work explaining when either might be expected to perform best and whether the benefits might be complementary.

In this chapter, we investigate the complementary benefits of self-training and contrastive pretraining. Interestingly, we find that the combination yields significant gains in UDA despite producing negligible gains in SSL. In experiments across eight distribution shift benchmarks (*e.g.* BREEDs (Santurkar et al., 2021), FMoW (Koh et al., 2021), Visda (Peng et al., 2017)), we observe that re-using unlabeled data for self-training (with FixMatch (Sohn et al., 2020)) after learning contrastive representations (with SwAV (Caron et al., 2020)), yields $> 5\%$ average improvement on OOD accuracy in UDA as compared to $< 0.8\%$ average improvement in SSL (Fig. 6.1).

Next, we aim to understand *why* the combination of self-training and contrastive learning is synergistic under distribution shift. To do so, we analyze both methods in a simplified distribution shift setting that models domain-independent or invariant, and domain-specific or spurious features. Our theoretical analysis highlights that: (i) under suitable augmentations contrastive pretraining on unlabeled data can learn a feature extractor that amplifies the invariant feature over the spurious (*feature amplification*); and (ii) self-training (ST) can learn the optimal target linear predictor, when initialized with a "good" classifier (learnt over contrastive features), thus improving *linear transferability.* We also show that contrastive pretrained features continue to be correlated with spurious features, and as a result the linear predictor (CL) learnt using source labeled data over these features is



(a)

(b)

Figure 6.1: *Self-training over Contrastive learning (STOC) improves over Contrastive Learning (CL) under distribution shift.* **(a)** We observe that in SSL settings, where labeled and unlabeled data are drawn from the same distribution, STOC offers negligible improvements over CL. In contrast, in UDA settings where there is distribution shift between labeled and unlabeled data, STOC offers gains over CL. Results aggregated across 8 benchmarks. Results on individual data in Table 6.1 and 6.2. **(b)** 2-D illustration of our simplified distribution setup, depicting decision boundaries learned by ERM and CL and how Self-Training (ST) updates those. ①, ②, and ③ summarize our theoretical results in Sec. 6.4.

suboptimal on target. Still, Cl outperforms source-only ERM in providing "good" initial pseudolabels on the target unlabeled data. Thus, self-training over the CL predictor (STOC) pretrained features unlearns any reliance on domain-dependent features and improves OOD performance relative to either method independently.

Finally, we connect our theoretical understanding of feature amplification done by contrastive learning, and improved linear transferability from self-training back to observed empirical gains. We linearly probe representations (fix representations and train only the linear head) learned by contrastive pretraining vs. no pretraining and find: (i) contrastive pretraining substantially improves the ceiling on the target accuracy (performance of optimal linear probe) compared to ERM; (ii) self-training mainly improves linear transfer, *i.e.* OOD performance for the linear probe trained with source labeled data.

6.2 Setup and Preliminaries

Task. Our goal is to learn a predictor that maps inputs $x \in \mathcal{X} \subseteq \mathbb{R}^d$ to outputs $y \in \mathcal{Y}$. We parameterize predictors $f = h \circ \Phi : \mathbb{R}^d \mapsto \mathcal{Y}$, where $\Phi : \mathbb{R}^d \mapsto \mathbb{R}^k$ is a feature map and $h \in \mathbb{R}^k$ is a classifier that maps the representation to the final scores or logits. Let $\mathrm{P}_s, \mathrm{P}_t$ be the source and target joint probability measures over $\mathcal{X} \times \mathcal{Y}$ with p_s and p_t as the corresponding probability density (or mass) functions. The distribution over unlabeled samples from both the union of source and target is denoted as $\mathrm{P}_{\mathsf{U}} = (1/2) \cdot \mathrm{P}_s(x) + (1/2) \cdot \mathrm{P}_t(x)$.

We study two particular scenarios: (i) Unsupervised Domain Adaptation (UDA); and (ii) Semi-Supervised Learning (SSL). In UDA, we assume that the source and target distributions have the same label marginals $\mathrm{P}_s(y) = \mathrm{P}_t(y)$ (*i.e.*, no label proportion shift) and the same Bayes optimal predictor, *i.e.*, $\arg\max_y p_s(y \mid x) = \arg\max_y p_t(y \mid x)$. We are given labeled samples from the source, and unlabeled pool from the target. In contrast in SSL, there is no distribution shift, *i.e.*, $\mathrm{P}_s = \mathrm{P}_t = \mathrm{P}_{\mathsf{U}}$. Here, we are given a small number of labeled examples and a comparatively large amount of unlabeled examples, both drawn from the same distribution, which we denote as P_t.

Unlabeled data is typically much cheaper to obtain, and our goal in both these settings is to leverage this along with labeled data to achieve good performance on the target distribution. In the UDA scenario, the challenge lies in generalizing out-of-distribution, while in SSL, the challenge is to generalize in-distribution despite the paucity of labeled examples. A predictor f is evaluated on distribution P via its accuracy, *i.e.*, $A(f, \mathrm{P}) = \mathbb{E}_{\mathrm{P}}(\arg\max f(x) = y)$.

Methods. We now introduce the algorithms used for learning from labeled and unlabeled data.

1. *Source-only ERM (ERM)*: A standard approach is to simply perform supervised learning on the labeled data by minimizing the empirical risk $\sum_{i=1}^{n} \ell(h \circ \Phi(x), y)$, for some classification loss $\ell : \mathbb{R} \times \mathcal{Y} \mapsto \mathbb{R}$ (*e.g.*, softmax cross-entropy) and labeled points $\{(x_i, y_i)\}_{i=1}^{n}$.

2. *Contrastive Learning (CL)*: We first use the unlabeled data to learn a feature extractor. In particular, the objective is to learn a feature extractor Φ_{cl} that maps augmentations (for e.g. crops or rotations) of the same input close to each other and far from augmentations of random other inputs (Caron et al., 2020; Chen et al., 2020a; Zbontar et al., 2021). Once we have Φ_{cl}, we learn a linear classifier h on top to minimize a classification loss on the labeled source data. We could either keep Φ_{cl} fixed or propagate gradients through.

   When clear from context, we also use CL to refer to just the contrastively pretrained backbone without training for downstream classification.

3. *Self-training (ST)*: This is a two-stage procedure, where the first stage performs source-only ERM by just looking at source-labeled data. In the second stage, we iteratively apply the current classifier on the unlabeled data to generate “pseudo-labels” and then update the classifier by minimizing a classification loss on the pseudolabeled data (Lee et al., 2013).

6.3 Self-Training Improves Contrastive Pretraining Under Distribution Shift

Self-Training Over Contrastive learning (STOC). Finally, rather than starting with a source-only ERM classifier, we propose to initialize self-training with a CL classifier, that was pretrained on unlabeled source and target data. ST uses that same unlabeled

Table 6.1: *Results in the UDA setup.* We report accuracy on target (OOD) data from which we only observe unlabeled examples during training. For benchmarks with multiple target distributions (*e.g.*, OH, Visda), we report avg accuracy on those targets. Results with source performance, individual target performance, and standard deviation numbers are in App. E.3.4.

Method	Living17	Nonliv26	Entity13	Entity30	FMoW (2 tgts)	Visda (2 tgts)	OH (3 tgts)	CIFAR→ CINIC	Avg
ERM	60.31	45.54	68.32	55.75	56.50	20.91	9.51	74.33	48.90
ST	71.29	56.79	77.93	66.37	56.79	38.03	10.47	78.19	56.98
CL	74.14	57.02	76.58	66.01	61.78	63.49	22.63	77.51	62.39
STOC (ours)	**82.22**	**62.23**	**81.84**	**72.00**	**65.25**	**70.08**	**27.12**	**79.94**	**67.59**

Table 6.2: *Results in the SSL setup.* We report accuracy on hold-out ID data. Recall that SSL uses labeled and unlabeled data from the same distribution during training. Refer to App. E.3.5 for ERM and ST.

Method	Living17	Nonliv26	Entity13	Entity30	FMoW	Visda	OH	CIFAR	Avg
CL	91.15	84.58	90.73	85.47	43.05	97.67	49.73	91.78	79.27
STOC (ours)	92.00	85.95	91.27	86.14	44.43	97.70	49.95	93.06	80.06

data again for pseudolabeling. As we demonstrate experimentally and theoretically, this combination of methods improves substantially over each independently.

Datasets. For both UDA and SSL, we conduct experiments across eight benchmark datasets: four BREEDs datasets (Santurkar et al., 2021)—Entity13, Entity30, Nonliving26, Living17; FMoW (Christie et al., 2018; Koh et al., 2021) from WILDS benchmark; Officehome (Venkateswara et al., 2017); Visda (Peng et al., 2017; 2018); and CIFAR-10 (Krizhevsky and Hinton, 2009). Each of these datasets consists of domains, enabling us to construct source-target pairs (e.g., CIFAR10, we consider CIFAR10→CINIC shift (Darlow et al., 2018)). In the UDA setup, we adopt the source and target domains standard to previous studies (details in App. E.3.2). Because the SSL setting lacks distribution shift, we do not need to worry about domain designations and default to using source alone. To simulate limited supervision in SSL, we sub-sample the original labeled training set to 10%.

Experimental Setup and Protocols. SwAV (Caron et al., 2020) is the specific algorithm that we use for contrastive pretraining. In all UDA settings, unless otherwise specified, we pool all the (unlabeled) data from the source and target to perform SwAV. For self-training, we apply FixMatch (Sohn et al., 2020), where the loss on source labeled data and on pseudolabeled target data are minimized simultaneously. For both methods, we fix the algorithm-specific hyperparameters to the original recommendations. For SSL settings, we perform SwAV and FixMatch on in-distribution unlabeled data. We experiment with Resnet18, Resnet50 (He et al., 2016) trained from scratch (*i.e.* random initialization). We do not consider off-the-shelf pretrained models (*e.g.*, on Imagenet (Russakovsky et al., 2015)) to avoid confounding our conclusions about contrastive pretraining. However, we note that our results on most datasets tend to be comparable to and sometimes exceed those obtained with ImageNet-pretrained models. For source-only ERM, as with other methods (FixMatch, SwAV), we default to using strong augmentation techniques: random horizontal flips, random crops, augmentation with Cutout (DeVries and Taylor, 2017), and RandAugment (Cubuk et al., 2020). Moreover, unless otherwise specified, we default to full finetuning with source-only ERM, both from scratch and after contrastive pretraining, and for ST with FixMatch. For UDA, given that the setup precludes access to labeled data from the target distribution, we use source hold-out performance to pick the best hyperparameters. During pretraining, early stopping is done according to lower values of pretraining loss. For more details on datasets, model architectures, and experimental protocols, see App. E.3[1].

Results on UDA setup. Both ST and CL individually improve over ERM across all datasets, with CL significantly performing better than ST on 5 out of 8 benchmarks (see Table 6.1). Even on datasets where ST is better than CL, their performance remains close. Combining ST and CL with STOC shows an 3–8% improvement over the best alternative, yielding an absolute improvement in average accuracy of 5.2%.

Note that by default, we train with CL on the combined unlabeled data from source and target. However, to better understand the significance of unlabeled target data in contrastive pretraining, we perform an ablation where the CL model was trained solely on unlabeled source data (refer to this as CL (source only); see App. E.3.4). We observe that ST on top of CL (source only) improves over ST (from scratch). However, the average performance of ST over CL (source only) is similar to that of standalone CL, maintaining an approximate 6% performance gap observed between CL and ST. This brings two key insights to the fore: (i) the observed benefit is not merely a result of the contrastive pretraining objective alone, but specifically CL with unlabeled target data helps; and (ii) both CL and ST leverage using target unlabeled data in a complementary nature.

Results on SSL setup. While CL improves over ST (as in UDA), unlike UDA, STOC doesn't offer any significant improvements over CL (see Table 6.2; ERM and ST results (refer to App. E.3.5). We conduct ablation studies with varying proportions of labeled data used for SSL, illustrating that there's considerable potential for improvement (see App. E.3.5). These findings highlight that the complementary nature of STOC over CL and ST individually is an artifact of distribution shift.

[1] For SwAV we use the code from `https://github.com/facebookresearch/swav`, and for self-training we use `https://github.com/acmi-lab/RLSbench`.

6.4 Theoretical Analysis and Intuitions

Our results on real-world datasets suggest that although self-training may offer little to no improvement over contrastive pretraining for in-distribution (*i.e.*, SSL) settings, it leads to substantial improvements when facing distribution shifts in UDA (Sec. **??**). Why do these methods offer complementary gains, but only under distribution shifts? In this section, we seek to answer this question by first replicating all the empirical trends of interest in a simple data distribution with an intuitive story (Sec. 6.4.1). In this toy model, we formally characterize the gains afforded by contrastive pretraining and self-training both individually (Secs. 6.4.2, 6.4.3) and when used together (Sec. 6.4.4).

Data distribution We consider binary classification and model the inputs as consisting of two kinds of features: $x = [x_{\text{in}}, x_{\text{sp}}]$, where $x_{\text{in}} \in \mathbb{R}^{d_{\text{in}}}$ is the invariant feature that is predictive of the label across both source P_s and target P_t and $x_{\text{sp}} \in \mathbb{R}^{d_{\text{sp}}}$ is the spurious feature that is correlated with the label y only on the source domain P_s but uncorrelated with label y in P_t. Formally, we sample $\mathrm{y} \sim \mathrm{Unif}\{-1, 1\}$ and generate inputs x conditioned on y as follows:

$$\begin{aligned} \mathrm{P}_s: \;\; & x_{\text{in}} \sim \mathcal{N}(\gamma \cdot \mathrm{y} w^\star, \Sigma_{\text{in}}) \;\; x_{\text{sp}} = \mathrm{y}\mathbf{1}_{d_{\text{sp}}} \\ \mathrm{P}_t: \;\; & x_{\text{in}} \sim \mathcal{N}(\gamma \cdot \mathrm{y} w^\star, \Sigma_{\text{in}}) \;\; x_{\text{sp}} \sim \mathcal{N}(\mathbf{0}, \Sigma_{\text{sp}}), \end{aligned} \tag{6.1}$$

where γ is the margin afforded by the invariant feature[2]. We set the covariance of the invariant features $\Sigma_{\text{in}} = \sigma_{\text{in}}^2 \cdot (\mathbf{I}_{d_{\text{in}}} - w^\star w^{\star\top})$. This makes the variance along the unknown predictive direction $w^\star$ to be zero. Note that the spurious feature is also completely predictive of the label in the source data. In fact, when d_{sp} is sufficiently large, x_{sp} is more predictive (than x_{in}) of y in the source. In the target, x_{sp} is distributed as a Gaussian with $\Sigma_{\text{sp}} = \sigma_{\text{sp}}^2 \mathbf{I}_{d_{\text{sp}}}$. We use $w_{\text{in}}{=}[w^\star, 0, ..., 0]^\top$ to refer to the invariant predictor (or direction), and $w_{\text{sp}} = [0, ..., 0, {}^{\mathbf{1}_{d_{\text{sp}}}}/\!_{\sqrt{d_{\text{sp}}}}]^\top$ for the spurious direction.

Data for UDA vs. SSL For convenience, whenever we have unlabeled data, we assume access to infinite unlabeled data and replace their empirical quantities with population counterparts. For SSL, we sample both finite labeled and infinite unlabeled data from the same distribution P_t, where spurious features are absent (to exclude easy-to-generalize features). For UDA, we further assume infinite labeled data from P_s (in addition to infinite unlabeled from P_t). Importantly, note that due to distribution shift, population access of P_s still captures the interesting aspects of distribution shifts—ERM on infinite labeled source data *does not* achieve optimal performance on target.

Methods and objectives Recall from Section 6.2 that we learn linear classifiers h over feature extractor Φ. For our toy setup, we consider linear feature extractors i.e. Φ is a matrix in $\mathbb{R}^{d \times k}$ and the prediction $f(x) = \mathrm{sgn}(h^\top \Phi x)$. We use the exponential loss $\ell(f(x), y) = \exp(-y f(x))$.

Self-training. ST performs ERM in the first stage using labeled data from the source, and then subsequently updates the head h by iteratively generating pseudolabels on the

[2] See App. E.4.1 for similarities and differences of our setup with prior works.

unlabeled target:

$$\begin{aligned}\mathcal{L}_{\text{st}}(h;\Phi) &:= \mathbb{E}_{\mathrm{P}_t(x)}\ell(h^\top \Phi x, \operatorname{sgn}(h^\top \Phi(x))) \\ \text{Update: } h^{t+1} &= \frac{h^t - \eta \nabla_h \mathcal{L}_{\text{st}}(h^t;\Phi)}{\|h^t - \eta \nabla_h \mathcal{L}_{\text{st}}(h^t;\Phi)\| 2}\end{aligned} \tag{6.2}$$

For ERM and ST, we train both h and Φ (equivalent to Φ being identity and training a linear head).

Contrastive pretraining. We obtain $\Phi_{\text{cl}} := \arg\min_\Phi \mathcal{L}_{\text{cl}}(\Phi)$ by minimizing the Barlow Twins objective (Zbontar et al., 2021), which prior works have shown is also equivalent to spectral contrastive and non-contrastive objectives (Cabannes et al., 2023; Garrido et al., 2022). Given probability distribution $\mathrm{P}_{\mathsf{A}}(a \mid x)$ for input x, and marginal P_{A}, we consider a constrained form of Barlow Twins in (6.3) which enforces features of "positive pairs" a_1, a_2 to be close while ensuring feature diversity. We assume a strict regularization ($\rho = 0$) for the theory arguments in the rest of the paper, and in App. E.4.2 we prove that all our claims hold for small ρ as well. For augmentations, we scale the magnitude of each co-ordinate uniformly by an independent amount, i.e., $a \sim \mathrm{P}_{\mathsf{A}}(\cdot \mid x) = \mathbf{c} \odot x$, where $\mathbf{c} \sim \text{Unif}[0,1]^d$. We try to mirror practical settings where the augmentations are fairly "generic", not encoding information about which features are invariant or spurious, and hence perturb all features symmetrically.

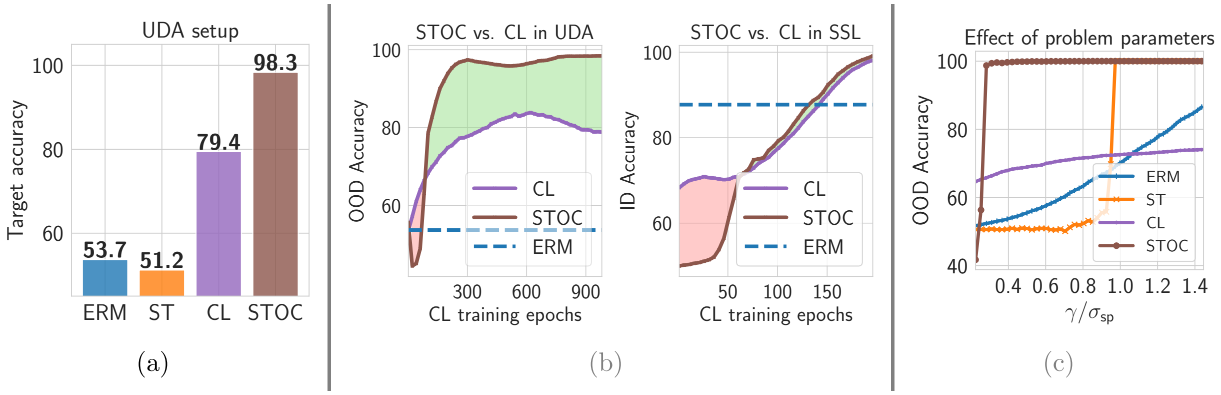
$$\begin{aligned}\mathcal{L}_{\text{cl}}(\Phi) &:= \mathbb{E}_{x \sim \mathrm{P}_{\mathsf{U}}} \mathbb{E}_{a_1, a_2 \sim \mathrm{P}_{\mathsf{A}}(\cdot|x)} \, \|\Phi(a_1) - \Phi(a_2)\|_2^2 \\ \text{s.t. } & \left\| \mathbb{E}_{a \sim \mathrm{P}_{\mathsf{A}}} \left[\Phi(a)\Phi(a)^\top\right] - \mathbf{I}_k \right\| F^2 \leqslant \rho\end{aligned} \tag{6.3}$$

Keeping the Φ_{cl} fixed, we then learn a linear classifier h_{cl} over Φ_{cl} to minimize the exponential loss on labeled source data (refer to as *linear probing*). For STOC, keeping the Φ_{cl} fixed and initializing the linear head with the CL linear probe (instead of source only ERM), we perform ST with (6.2).

Example 6.4.1. *For the setup in* (12.1)*, we choose* $\gamma = 0.5$*,* $\sigma_{\text{sp}}^2 = 1.$*, and* $\sigma_{\text{in}}^2 = 0.05$ *with* $d_{\text{in}} = 5$ *and* $d_{\text{sp}} = 20$ *for our running example.* $\gamma/\sqrt{d_{\text{sp}}}$ *controls signal to noise ratio in the source such that spurious feature is easy-to-learn and the invariant feature is harder-to-learn. Here,* σ_{sp} *controls the noise in target which we show later is critical in unlearning the spurious feature with CL.*

6.4.1 Simulations and Intuitive Story: A Comparative Study Between SSL and DA

Our setup captures real-world trends in UDA setting. Our toy setup (in Example 6.4.1) accentuates the behaviors observed on real-world datasets (Fig. 6.2(a)): (i) both ERM and ST yield close to random performance (though ST performs slightly worse than ERM); (ii) CL improves over ERM but still yields sub-optimal target performance; (iii) STOC then further improves over CL, achieving near-optimal target performance. Note that, a linear predictor can improve target performance only by reducing its dependence on spurious feature x_{sp}, and increasing it on invariant feature x_{in} (along $w^\star$). Given this, we



(a)

(b)

(c)

Figure 6.2: *Our simplified model of shift captures real-world trends and theoretical behaviors:* **(a)** Target (OOD) accuracy separation in the UDA setup (for problem parameters in Example 6.4.1). **(b)** Comparison of the benefits of STOC (ST over CL) over just CL in UDA and SSL settings, done across training iterations for contrastive pretraining. **(c)** Comparison between different methods in UDA setting, as we vary problem parameters γ and σ_{sp}, connecting our theory results in Sec. 6.4.

can explain our trends if we understand the following: (i) how ST reduces dependence on spurious feature when done after CL; (ii) why CL helps reduce but not completely eliminate the reliance of linear head on spurious features. Before we present intuitions, we ablate over a key problem parameter that affects both the target performance and conditions for ST to work.

An intuitive story. We return to the question of why self-training improves over contrastive learning under distribution shift in our Example 6.4.1. When the classifier at initialization of ST relies more on spurious features, ST aggravates this dependency. However, as the problem becomes easier (with increasing $\gamma/\sigma_{\mathrm{sp}}$), the source-only ERM classifier will start relying more on invariant rather than spurious feature. Once this ERM classifier is sufficiently accurate on the target, ST unlearns any dependency on spurious features achieving optimal target performance. This is because the initial pseudolabels on target unlabeled data are sufficiently accurate for self-training to improve *linear transferability*. In contrast, we observe that CL performs better than ERM since contrastive pretraining learns a feature map that is correlated more with the invariant than the spurious feature. This implies that CL does *feature amplification*: decreasing reliance on spurious features (as compared to ERM), but doesn't completely eliminate them, thereby remaining sub-optimal on target. Combining ST and CL, a natural hypothesis explaining our trends is that CL provides a favorable initialization (through feature amplification) for ST to now improve linear transferability.

Effect of $\gamma/\sigma_{\mathrm{sp}}$ on success of ST. Our intuitive understanding is reinforced by our experiment that increases the ratio of margin γ and variance of spurious feature on target σ_{sp} (keeping others constant). Doing this makes the problem becomes easier because γ directly affects the signal on x_{in} and reducing σ_{sp} helps ST to unlearn x_{sp} (see App. E.4.3). In Fig. 6.2(c), we see that a phase transition occurs for ST, *i.e.*, after a certain threshold of ${}^{\gamma}/_{\sigma_{\text{sp}}}$, ST successfully recovers the optimal target predictor. This hints that ST has a binary effect, where beyond a certain magnitude of ${}^{\gamma}/_{\sigma_{\text{sp}}}$, ST can amplify the signal on domain invariant feature to obtain optimal target predictor. This explains the ability of ST to improve linear transferability when the initial classifier has sufficiently low target error. On the other hand, the performance of CL and ERM improve gradually where CL achieves high performance due to feature amplification, which occurs at even small ratios of ${}^{\gamma}/_{\sigma_{\text{sp}}}$. One way of viewing this trend with CL is that it magnifies the effective ${}^{\gamma}/_{\sigma_{\text{sp}}}$ in its representation space, because of which a linear head trained over these representations has a good performance at low values of the ratio. Consequently, the *phase transition* of STOC occurs much sooner then that of ST. Finally, we note that for CL the rate of performance increase diminishes at high values of ${}^{\gamma}/_{\sigma_{\text{sp}}}$ because CL fails to reduce dependency along x_{sp} beyond a certain point.

Why disparate behaviors for out-of-distribution vs. in distribution? In the SSL setup, recall, there is no distribution shift. In Example 6.4.1, we sample $50k$ unlabeled data and 100 labeled data from the same (target) distribution to simulate SSL setup. Substantiating our findings on real-world data, we observe that STOC provides a small to negligible improvement over CL (refer to App. E.4). To understand why such disparate behaviors emerge, recall that in the UDA setting, the main benefit of STOC lies in picking up reliance on "good" features for OOD data, facilitated by CL initialization. While contrastive pretraining uncovers features that are "good" for OOD data, it also learns more predictive source-only features (which are not predictive at all on target). As a result, linear probing with source-labeled data picks up these source-only features, leaving considerable room for improvement on OOD data with further self-training. On the other hand, in the SSL setting, the limited ID labeled data might provide enough signal to pick up features predictive on ID data, leaving little to no room for improvement for further self-training. Corroborating our intuitions, throughout the CL training in the toy setup, when CL doesn't achieve near-perfect generalization, the improvements provided by STOC for each checkpoint remain minimal. On the other hand, for UDA setup, after reaching a certain training checkpoint in CL, STOC yields significant improvement (Fig. 6.2(b)).

In the next sections, we formalize our intuitions and analyze why ST and CL offer complementary benefits when dealing with distribution shifts. Formal statements and proofs are in App. E.5.

6.4.2 Conditions for Success and Failure of Self-training over ERM from Scratch

In our results on Example 6.4.1, we observe that performing ST after ERM yields a classifier with near-random target accuracy. In Theorem 6.4.2, we characterize conditions under which ST fails and succeeds.

Theorem 6.4.2 (Informal; Conditions for success and failure of ST over ERM)**.** *The target accuracy of ERM classifier, is given by* $0.5 \cdot \operatorname{erfc}\left(-{}^{\gamma^2}/(\sqrt{2d_{\text{sp}}}\cdot\sigma_{\text{sp}})\right)$*. Then ST performed in the second stage yields: (i) a classifier with* ≈ 0.5 *target accuracy when* $\gamma < {}^{1}/_{2\sigma_{\text{sp}}}$ *and* $\sigma_{\text{sp}} \geqslant 1$*;*

and (ii) a classifier with near-perfect target accuracy when $\gamma \geqslant \sigma_{\mathrm{sp}}$.

The informal theorem above abstracts the exact dependency of $\gamma, \sigma_{\mathrm{sp}}$, and d_{sp} for the success and failure of ST over ERM. Our analysis highlights that while ERM learns a perfect predictor along w_{in} (with norm γ), it also learns to depend on w_{sp} (with norm $\sqrt{d_{\mathrm{sp}}}$) because of the perfect correlation of x_{sp} with labels on the source. Our conditions depict that when the $\gamma/\sigma_{\mathrm{sp}}$ is sufficiently small, then ST continues to erroneously enhance its reliance on the x_{sp} feature for target prediction, resulting in near-random target performance. Conversely, when $\gamma/\sigma_{\mathrm{sp}}$ is larger than 1, the signal in x_{in} is correctly used for predictor on the majority of target points, and ST eliminates the x_{sp} dependency, converging to an optimal target classifier.

Our proof analysis shows that if the ratio of the norm of the classifier along in the direction of $w^\star$ is smaller than w_{sp} by a certain ratio then the generated pseudolabels (incorrectly) use x_{sp} for its prediction further increasing the component along w_{sp}. Moreover, normalization further diminishes the reliance along $w^\star$, culminating in a near-random performance. The opposite occurs when the ERM classifier achieves a signal along $w^\star$ that is sufficiently stronger than along w_{sp}. Upon substituting the parameters used in Example 6.4.1, the ERM and ST performances as determined by Theorem 6.4.2 align with our empirical results, notably, ST performance on target being near-random.

6.4.3 CL Captures Both Features But Amplifies Invariant Over Spurious Features

Here we show that minimizing the contrastive loss (6.3) on unlabeled data from both P_s and P_t gives us a feature extractor Φ_{cl} that has a higher inner product with the invariant feature over the spurious feature. First, we derive a closed form expression for Φ_{cl} that holds for any linear backbone and augmentation distribution. Then, we introduce assumptions on the augmentation distribution (or equivalently on $w^\star$) and other problem parameters, that are sufficient to prove amplification.

Proposition 6.4.3 (Barlow Twins solution)**.** *The solution for* (6.3) *is $U_k^\top \Sigma_{\mathsf{A}}^{-1/2}$ where U_k are the top k eigenvectors of $\Sigma_{\mathsf{A}}^{-1/2}\, \widetilde{\Sigma}\, \Sigma_{\mathsf{A}}^{-1/2}$. Here, $\Sigma_{\mathsf{A}} := \mathbb{E}_{a \sim \mathrm{P}_{\mathsf{A}}}[aa^\top]$ is the covariance over augmentations, and $\widetilde{\Sigma} := \mathbb{E}_{x \sim \mathrm{P}_{\mathsf{U}}}[\widetilde{a}(x)\widetilde{a}(x)^\top]$ is the covariance matrix of mean augmentations $\widetilde{a}(x) := \mathbb{E}_{\mathrm{P}_{\mathsf{A}}(a|x)}[a]$.*

The above result captures the effect of augmentations through the matrix U_k. If there were no augmentations, then $\Sigma_{\mathsf{A}} = \widetilde{\Sigma}$, implying that U_k could then be any random orthonormal matrix. On the other hand if augmentation distributions change prevalent covariances in the data, *i.e.*, Σ_{A} is very different from $\widetilde{\Sigma}$, the matrix U_k would bias the CL solution towards directions that capture significant variance in marginal distribution on augmented data, but have low conditional variance, when conditioned on original point x—precisely the directions with low invariance loss. Hence, we can expect that CL would learn components along both invariant w_{in} and spurious w_{sp} because: (i) these directions explain a large fraction of variance in the raw data; (ii) augmentations that randomly scale down dimensions would add little variance along w_{sp} and w_{in} compared to noise directions in their null space. On the other hand it is unclear which of these directions is amplified more in Φ_{cl}. The following assumption and amplification result conveys that when the noise in target (σ_{sp}) is suficiently large, the CL solution amplifies the invariant feature over the spurious feature.

Assumption 1 (Informal; Alignment of $w^\star$ with augmentations)**.** *We assume that $w^\star$ aligns with $\mathrm{P}_{\mathsf{A}}(\cdot \mid x)$, i.e., $\forall x$, $\mathbb{E}_{a|x}[a^\top w^\star] = {}^1\!/_2 \cdot x^\top \mathrm{diag}(\mathbb{1}_d) w^\star$ is high. Hence, we assume $w^\star = {}^{\mathbb{1}_{d_{\text{in}}}}\!/_{\sqrt{d_{\text{in}}}}$.*

One implication of Assumption 1 is that when $w^\star = {}^{\mathbb{1}_{d_{\text{in}}}}\!/_{\sqrt{d_{\text{in}}}}$, only the top two eigenvectors lie in the space spanned by w_{in} and w_{sp}. To analyze our amplification with fewer eigenvectors from Proposition 6.4.3 while retaining all relevant phenomena, we assume $w^\star = {}^{\mathbb{1}_{d_{\text{in}}}}\!/_{\sqrt{d_{\text{in}}}}$ for mathematical convenience. While Assumption 1 permits a tighter theoretical analysis, our empirical results in Sec. 6.4.1 hold more generally for $w^\star \sim \mathcal{N}(0, \mathbf{I}_{d_{\text{in}}})$.

Theorem 6.4.4 (Informal; CL recovers both w_{in} and w_{sp} but amplifies w_{in})**.** *Under Assumption 1, the CL solution Φ_{cl}=$[\phi_1, \phi_2, ..., \phi_k]$ satisfies $\phi_j^\top w_{\text{in}} = \phi_j^\top w_{\text{sp}} = 0 \;\forall j \geqslant 3$, $\phi_1 = c_1 w_{\text{in}} + c_3 w_{\text{sp}}$ and $\phi_2 = c_2 w_{\text{in}} + c_4 w_{\text{sp}}$. For constants $K_1, K_2 > 0$, $\gamma = {}^{K_1 K_2}\!/_{\sigma_{\text{sp}}}$, $d_{\text{sp}} = {}^{\sigma_{\text{sp}}^2}\!/_{K_2^2}$, $\forall \epsilon > 0$, $\exists \sigma_{\text{sp}_0}$, such that for $\sigma_{\text{sp}} \geqslant \sigma_{\text{sp}_0}$, $\left|{}^{c_1}\!/_{c_3} - {}^{K_1 K_2^2 d_{\text{in}}}\!/_{2L\sigma_{\text{in}}^2(d_{\text{in}}-1)}\right| \leqslant \epsilon$, and $\left| \left|{}^{c_2}\!/_{c_4}\right| - {}^{L\sqrt{d_{\text{sp}}}}\!/_{\gamma} \right| \leqslant \epsilon$, where $L = 1 + K_2^2$.*

We analyze the amplification of $w_{\text{in}}/w_{\text{sp}}$ with contrastive learning in the regime where σ_{sp} is large enough. In other words, if the target distribution has sufficient noise along the spurious feature, the augmentations prevent the CL solution from extracting components along w_{sp}. Thus, in our analysis, we first analyze the amplification factors asymptotically ($\sigma_{\text{sp}} \to \infty$), and then use the asymptotic behavior to draw conclusions for the regime where σ_{sp} is large but finite.

Theorem 6.4.4 conveys two results: (i) CL recovers components along both w_{in} and w_{sp} through ϕ_1, ϕ_2; and (ii) it increases the norm along w_{in} more than w_{sp}. The latter is evident because the margin separating labeled points along w_{in} is now amplified by a factor of $\left|{}^{c_2}\!/_{c_4}\right| = \Omega({}^{L\sqrt{d_{\text{sp}}}}\!/_{\gamma})$ in ϕ_2. Naturally, this will improve the target performance of a linear predictor trained over CL representations. At the same time, we also see that in ϕ_1, the component along w_{sp} is still significant (${}^{c_1}\!/_{c_3} = \mathcal{O}({}^1\!/_{L\sigma_{\text{in}}^2})$). Intuitively, CL prefers the invariant feature since augmentations amplify the noise along w_{sp} in the target domain. At the same time, the variance induced by augmentations along w_{sp} in source is still very small due to which the dependence on w_{sp} is not completely alleviated. Due to the remaining components along w_{sp}, the target performance for CL can remain less than ideal. Both the above arguments on target performance are captured in Corollary 6.4.5.

Corollary 6.4.5 (Informal; CL improves OOD error over ERM but is still imperfect)**.** *For $\gamma, \sigma_{\text{sp}}, d_{\text{sp}}$ defined as in Theorem 6.4.4, $\exists \sigma_{\text{sp}_1}$ such that for all $\sigma_{\text{sp}} \geqslant \sigma_{\text{sp}_1}$, the target accuracy of CL (linear predictor on Φ_{cl}) is $\geqslant 0.5\,\mathrm{erfc}\left(-L' \cdot {}^{\gamma}\!/_{\sqrt{2}\sigma_{\text{sp}}}\right)$ and $\leqslant 0.5\,\mathrm{erfc}\left(-4L' \cdot {}^{\gamma}\!/_{\sqrt{2}\sigma_{\text{sp}}}\right)$, where $L' = {}^{K_2^2 K_1}\!/_{\sigma_{\text{in}}^2(1-{}^1\!/_{d_{\text{in}}})}$. When $\sigma_{\text{sp}_1} > \sigma_{\text{in}}\sqrt{1 - {}^1\!/_{d_{\text{in}}}}$, the lower bound on accuracy is strictly better than ERM from scratch.*

While Φ_{cl} is still not ideal for linear probing, in the next part we will see how Φ_{cl} can instead be sufficient for subsequent self-training to unlearn the remaining components along spurious features.

6.4.4 Improvements with Self-training Over Contrastive Learning

The result in the previous section highlights that while CL may improve over ERM, the linear probe continues to depend on the spurious feature. Next, we characterize the behavior STOC. Recall, in the ST stage, we iteratively update the linear head with (6.2) starting with the CL backbone and head.

Theorem 6.4.6 (Informal; ST improves over CL)**.** *Under the conditions of Theorem 6.4.4 and* $d_{\text{sp}} \leqslant K_1^2 \cdot K_2^{2/3}$*, the target accuracy of ST over CL is lower bounded by* $0.5 \cdot \text{erfc}\left(-\left|{}^{c2}/_{c4}\right| \cdot {}^{\gamma}/_{(\sqrt{2}\sigma_{\text{sp}})}\right) \approx 0.5 \cdot \text{erfc}\left(-L\sqrt{{}^{d_{\text{sp}}}}/_{(\sqrt{2}\sigma_{\text{sp}})}\right)$ *where* c_2 *and* c_4 *are the coefficients of feature* ϕ_2 *along* w_{in} *and* w_{sp} *learned by BT.*

The above theorem states that when $\sqrt{d_{\text{sp}}}/\sigma_{\text{sp}} \gg 1$ the target accuracy of ST over CL is close to 1. In Example 6.4.1, the lower bound of the accuracy of ST over CL is $\text{erfc}\left(-\sqrt{10}\right) \approx 2$ showing near-perfect target generalization. Recall that Theorem 6.4.5 shows that CL yields a linear head that mainly depends on both the invariant direction w_{in} and the spurious direction w_{sp}. At initialization, the linear head trained on the CL backbone has negligible dependence on ϕ_2 (under conditions in Theorem 6.4.5). Building on that, the analysis in Theorem 6.4.6 captures that ST gradually reduces the dependence on w_{sp} by learning a linear head that has a larger reliance on ϕ_2, which has a higher "effective" margin on the target, thus increasing overall dependency on w_{in}.

Theoretical comparison with SSL. Our analysis until now shows that linear probing with source labeled data during CL picks up features that are more predictive of source label under distribution shift, leaving a significant room for improvement on OOD data when self-trained further. In UDA, the primary benefit of ST lies in picking up the features with a high "effective" margin on target data that are not picked up by linear head trained during CL. In contrast, in the SSL setting, the limited ID labeled data may provide enough signal in picking up high-margin features that are predictive on ID data, leaving little to no room for improvement for further ST. We formalize this intuition in App. E.5 when the CL/ERM predictors are trained with margin based surrogate losses for learning the classifier.

6.4.5 Reconciling Practice: Implications for Deep Non-Linear Networks

In this section, we experiment with deep non-linear backbone (*i.e.*, Φ_{cl}). When we continue to fix Φ_{cl} during CL and STOC, the trends we observed with linear networks in Sec. 6.4.1 continue to hold. We then perform full fine-tuning with CL and STOC, i.e., propagate gradients even to Φ_{cl}, as commonly done in practice. We present key takeaways here but detailed experiments are in App. E.4.4.

Benefits of augmentation for self-training. ST while updating Φ_{cl} can hurt due to overfitting issues when training with the finite sample of labeled and unlabeled data (drop by >10% over CL). This is due to the ability of deep networks to overfit on confident but

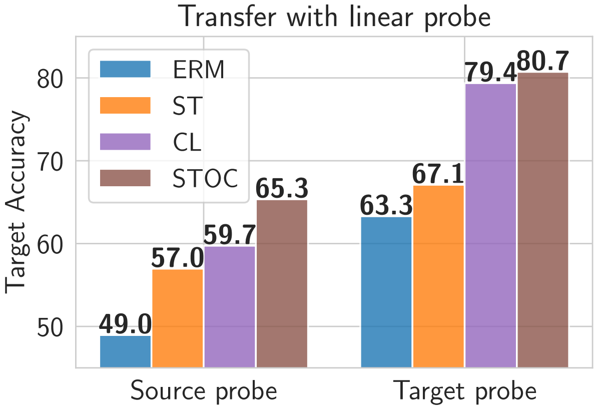


Figure 6.3: *Target accuracy with source and target linear probes*, which freezes backbones trained with various objectives and trains only the head in UDA setup. Avg. accuracy across all datasets. We observe that: (i) ST improves the linear transferability of source probes, and (ii) CL improves representations.

incorrect pseudolabels on target data (Zhang et al., 2017). This exacerbates components along w_{sp} and we find that augmentations (and other heuristics) typically used in practice (*e.g.* in FixMatch (Sohn et al., 2020)) help avoid overfitting on incorrect pseudolabels.

Can ERM and ST over contrastive pretraining improve features? We find that self-training can also slightly improve features when we update the backbone with the second stage of STOC and when the CL backbone is early stopped sub-optimally (*i.e.* at an earlier checkpoint in Fig. 6.2(b)). This feature finetuning can now widen the gap between STOC and CL in SSL settings, as compared to the linear probing gap (as in 6.2). This is because STOC can now improve performance beyond just recovering the generalization gap for the linear head (which is typically small). However, STOC benefits are negligible when CL is not early stopped sub-optimally, *i.e.*, trained till convergence. Thus, it remains unclear if STOC and CL have complementary benefits for feature learning in UDA or SSL settings. Investigating this is an interesting avenue for future work.

6.5 Connecting Experimental Gains with Theoretical Insights

Our theory emphasizes that under distribution shift contrastive pretraining does feature amplification which effectively improves the representations for target data, while self-training primarily improves linear transferability for the classifier learned on top of CL features. To investigate different methods in our UDA setup, we study the representations learned by each of them. We fix the representations and train linear heads over them to answer two questions: (i) How good are the representations in terms of their *ceiling* of target accuracy (performance of the optimal linear probe)?—we evaluate this by training the classifier head on target labeled data (*i.e.*, target linear probe); and (ii) How well do heads trained on source generalize to target?—we assess this by training a head on source labeled data (source linear probe) and evaluate its difference with target linear probe. For both, we plot target accuracy. We make two *intriguing* observations Fig. 6.3):

Does CL improve representations over ERM features? Yes. We observe a substantial difference in accuracy ($\approx$ 14% gap) of target linear probes on backbones trained with contrastive pretraining (*i.e.* CL, STOC) and without it (*i.e.*, ERM, ST) highlighting that CL significantly pushes the performance ceiling over non-contrastive features. As a side, our findings also stand in contrast to recent studies suggesting that ERM features might be "good enough" for OOD generalization (Kirichenko et al., 2022; Rosenfeld et al., 2022). Instead, the observed gains with contrastively pretrained backbones (*i.e.* CL, STOC) demonstrate that target unlabeled data can be leveraged to further improve over ERM features.

Do CL features yield *perfect* linear transferability from source to target? Recent works (HaoChen et al., 2022; Shen et al., 2022) conjecture that under certain conditions CL representations, linear probes learned with source labeled data may transfer perfectly from source to target. However, we observe that this doesn't hold strictly in practice, and in fact, the linear transferability can be further improved with ST. We first note a significant gap between the performance of source linear probes and target linear probes illustrating that linear transferability is not perfect in practice. Moreover, while the accuracy of target linear probes doesn't change substantially between CL and STOC, the accuracy of the source linear probe improves significantly. Similar observations hold for ERM and ST, methods trained without contrastive pretraining. This highlights that ST performs "feature refinement" to improve source to target linear transfer (with relatively small improvements in their respective target probe performance). *The findings highlight the complementary nature of benefits on real-world data: ST improves linear transferability while CL improves representations.*

6.6 Connections to Prior Work

Our empirical results and our analyses offer a perspective that contrasts with the prior literature that argues for the individual optimality of contrastive pretraining and self-training. We outline the key differences from existing studies here, and delve into other related works in App. E.1.

Limitations of prior work analyzing contrastive learning Prior works (HaoChen et al., 2022; Shen et al., 2022) analyzing CL first make assumptions on the consistency of augmentations with labels (Cabannes et al., 2023; HaoChen et al., 2021; Johnson et al., 2022; Saunshi et al., 2022), and specifically for UDA make stronger ones on the augmentation graph connecting examples from same domain or class more than cross-class/cross-domain ones. While this is sufficient to prove linear transferability, it is unclear if this holds in practice when augmentations are imperfect, *i.e.* if they fail to mask the spurious features completely—as corroborated by our findings in Sec. 6.4.5. We show why this also fails in our simplified setup in App. E.6.1.

Limitations of prior work analyzing self-training Prior research views self-training as consistency regularization, ensuring pseudolabels for original samples align with their augmentations (Cai et al., 2021a; Sohn et al., 2020; Wei et al., 2020). This approach abstracts the role played by the optimization algorithm and instead evaluates the global minimizer of a population objective promoting pseudolabel consistency. It also relies on specific assumptions about class-conditional distributions to guarantee pseudolabel accuracy across domains. However, this framework doesn't address issues in iterative label propagation. For example, when augmentation distribution has long tails, the consistency of pseudolabels depends on the sampling frequency of "favorable" augmentations (for more discussion see App. E.6.2). Our analysis thus follows the iterative examination of self-training (Chen et al., 2020b).

6.7 Conclusion

In this study, we highlight the synergistic behavior of self-training improving linear transferability and contrastive pretraining learning more "invariant" features under distribution shift. Shifts in distribution are commonplace in real-world applications of machine learning, and even under natural, non-adversarial distribution shifts, the performance of machine learning models often drops. By simply combining existing techniques in self-training and constrastive learning, we find that we can improve accuracy by 3–8% rather than using either approach independently. Despite these significant improvements, we note that one limitation of this combined approach is that performing self-training sequentially after contrastive pretraining increases the computation cost for UDA. The potential for integrating these benefits into one unified training paradigm is yet unclear, presenting an interesting direction for future exploration.

Beyond this, we note that our theoretical framework primarily confines the analysis to training the backbone and linear network independently during the pretraining and fine-tuning/self-training phases. Although our empirical observations apply to deep networks with full fine-tuning, we leave a more rigorous theoretical study of full fine-tuning for future work. Our theory also relies on a covariate shift assumption (where we assume that label distribution also doesn't shift). Investigating the complementary nature of self-training and contrastive pretraining beyond the covariate shift assumption would be another interesting direction for future work.

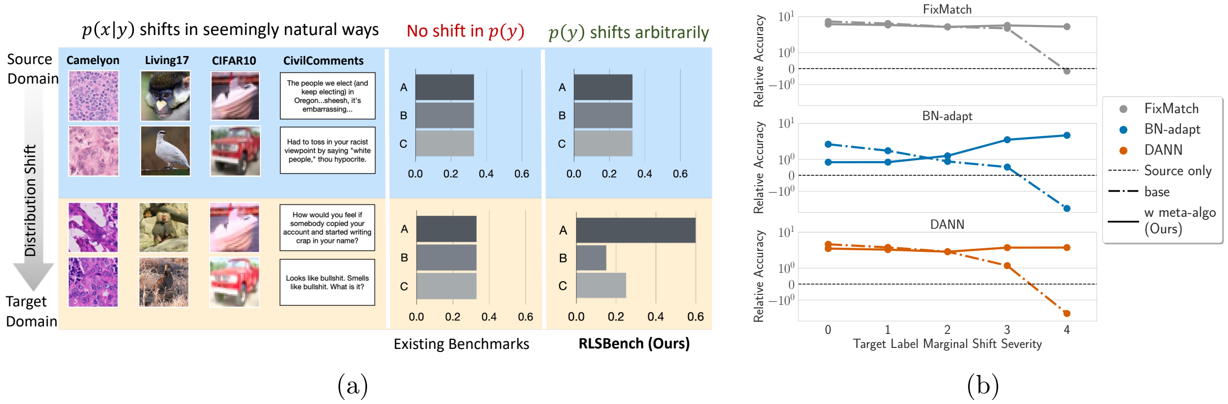
Chapter 7

RLSbench: Domain Adaptation Under Relaxed Label Shift

Based on Garg et al. (2023a): Saurabh Garg, Nick Erickson, James Sharpnack, Alex Smola, Sivaraman Balakrishnan, and Zachary Chase Lipton. Rlsbench: Domain adaptation under relaxed label shift. In International Conference on Machine Learning, 2023.

Abstract

In this chapter, we will combine natural input perturbations and label distribution shifts. Despite the emergence of principled methods for domain adaptation under label shift, their sensitivity to shifts in class conditional distributions is precariously underexplored. We introduce RLSBENCH, a large-scale benchmark for *relaxed label shift*, consisting of >500 distribution shift pairs spanning vision, tabular, and language modalities, with varying label proportions. Unlike existing benchmarks, which primarily focus on shifts in class-conditional $p(x|y)$, our benchmark also focuses on label marginal shifts. First, we assess 13 popular domain adaptation methods, demonstrating more widespread failures under label proportion shifts than were previously known. Next, we develop an effective two-step meta-algorithm that is compatible with most domain adaptation heuristics: (i) *pseudo-balance* the data at each epoch; and (ii) adjust the final classifier with a target label distribution estimate. The meta-algorithm improves existing domain adaptation heuristics under large label proportion shifts, often by 2–10% in accuracy, while having a minimal negative effect in the worst-case (<0.5%) when label proportions do not shift. We hope that these findings and the availability of RLSBENCH will encourage researchers to rigorously evaluate proposed methods in relaxed label shift settings. Code is publicly available at https://github.com/acmi-lab/RLSbench.



(a)

(b)

Figure 7.1: *Domain adaptation under Relaxed Label Shift.* **(a) Overview of RLSbench setup:** Unlike existing benchmarks for which the label marginal $p(y)$ doesn't shift, in RLSbench, $p(y)$ can shift arbitrarily. The class conditionals $p(x|y)$ shift in seemingly natural ways following popular benchmarks. RLSbench draws on 14 multi-domain datasets spanning vision, NLP, and tabular modalities. **(b) Key results:** As the severity of target label proportion increases, the performance of existing popular DA methods degrades, often dropping below source-only classifiers. DA methods, when paired with our meta-algorithm, significantly improve over a source-only classifier.

7.1 Introduction

In this chapter, we develop RLSBENCH, the first standardized test bed of *relaxed label shift* settings, where $p(y)$ can shift arbitrarily and the class conditionals $p(x|y)$ can shift in seemingly natural ways (following the popular DA benchmarks). While existing DA benchmarks typically focus on shifts in $p(x|y)$, our benchmarks additionally focuses on shifts in label marginals $p(y)$. We evaluate a collection of popular DA methods based on domain-invariant representation learning, self-training, and test-time adaptation across 14 multi-domain datasets spanning vision, Natural Language Processing (NLP), and tabular modalities. The different domains in each dataset present a different shift in $p(x|y)$. Since these datasets exhibit minor to no shift in label marginal, we simulate shift in target label marginal via stratified sampling with varying severity. Overall, we obtain 560 different source and target distribution shift pairs and train $> 30k$ models in our testbed.

Based on our experiments on RLSBENCH, we make several findings. First, we observe that while popular DA methods often improve over a source-only classifier absent shift in target label distribution, their performance tends to degrade, dropping below source-only classifiers under severe shifts in target label marginal. Next, we develop a meta-algorithm with two simple corrections: (i) re-sampling the data to balance the source and pseudo-balance the target; (ii) re-weighting the final classifier using an estimate of the target label marginal. We observe that in these relaxed label shift settings, the performance of existing DA methods (e.g. CDANN, FixMatch, and BN-adapt), when paired with our meta-algorithm, significantly improves over a source-only classifier. On the other hand, existing methods specifically proposed for relaxed label shift (e.g., IW-CDANN and SENTRY), often fail to improve over a source-only classifier and significantly underperform when compared to existing DA methods paired with our meta-algorithm.

Overall, RLSbench provides a comprehensive and standardized suite for label distributions shifts, bringing existing benchmarks one step closer to exhibit the sort of diversity that we should expect to encounter when deploying models in the wild. Our findings emphasize the effectiveness of a simple, previously overlooked baseline. We hope that the RLSbench and our meta-algorithm (that can be paired with any DA method) provide a framework for rigorous and reproducible future research in relaxed label shift scenarios.

7.2 Preliminaries and Prior Work

This chapter focuses on the *relaxed label shift* setting. In particular, we assume that the label distribution can shift from source to target arbitrarily but that $p(x|y)$ varies between source and target in some comparatively restrictive way (e.g., shifts arising naturally in the real-world like ImageNet (Russakovsky et al., 2015) to ImageNetV2 (Recht et al., 2019b)). Mathematically, we assume a divergence-based restriction on $p(x|y)$. That is, for some small $\epsilon > 0$ and distributional distance $\mathcal{D}$, we have $\max_y \mathcal{D}(p_s(x|y), p_t(x|y)) \leqslant \epsilon$ and allow an arbitrary shift in the label marginal $p(y)$. We discuss several precise instantiations in App. F.7. However, in practice, it's hard to empirically verify these distribution distances for small enough ϵ with finite samples. Moreover, we lack a rigorous characterization of the sense in which those shifts arise in popular DA benchmarks, and since, the focus of our work is on the empirical evaluation with real-world datasets, we leave a formal investigation for future work.

The goal in DA is to adapt a predictor from a source distribution with labeled data to a target distribution from which we only observe unlabeled examples. While prior work addressing relaxed label shift has primarily focused on classifier performance, we also separately evaluate methods for estimating the target label marginal. This can be beneficial for two reasons. First, it can shed more light into how improving the estimates of target class proportion improves target performance. Second, understanding how the class proportions are changing can be of independent interest.

7.2.1 Prior Work

Relaxed Label Shift Exploring the problem of shift in label marginal from source to target with natural variations in $p(x|y)$, a few papers highlighted theoretical and empirical failures of DA methods based on domain-adversarial neural network training (Johansson et al., 2019; Wu et al., 2019; Yan et al., 2017; Zhao et al., 2019). Subsequently, several papers attempted to handle these problems in domain-adversarial training (Liu et al., 2021b; Manders et al., 2019; Prabhu et al., 2021; Tachet et al., 2020; Tan et al., 2020). However, these methods often lack comparisons with other prominent DA methods and are evaluated under different datasets and model selection criteria. To this end, we perform a large scale rigorous comparison of popular representative DA methods in a standardized evaluation framework.

Distinction from previous distribution shift benchmark studies Previous studies evaluating robustness under distribution shift predominantly focuses on transfer learning and domain generalization settings Djolonga et al. (2021); Gulrajani and Lopez-Paz (2020); Koh et al. (2021); Wenzel et al. (2022); Wiles et al. (2021). Hendrycks et al. (2021b); Taori et al. (2020) studies the impact of robustness interventions (e.g. data augmentation techniques, adversarial training) on target (out of distribution) performance. Notably, Sagawa et al. (2021) focused on evaluating DA methods on WILDS-2.0. Our work is complementary to these studies, as we present the first extensive study of DA methods under shift in $p(y)$ and natural variations in $p(x|y)$.

7.3 RLSBENCH: A Benchmark for Relaxed Label Shift

In this section, we introduce RLSBENCH, a suite of datasets and DA algorithms that are at the core of our study. Motivated by correction methods for the (stricter) label shift setting (Lipton et al., 2018b; Saerens et al., 2002) and learning under imbalanced datasets (Cao et al., 2019a; Wei et al., 2021), we also present a meta-algorithm with simple corrections compatible with almost any DA method.

7.3.1 Datasets

RLSBENCH builds on 14 multi-domain datasets for classification, including tasks across applications in object classification, satellite imagery, medicine, and toxicity detection. Across these datasets, we obtain a total of 56 different source and target pairs. More details about datasets are in App. F.4.

(i) **CIFAR-10** which includes the original CIFAR-10 (Krizhevsky and Hinton, 2009), CIFAR-10-C (Hendrycks and Dietterich, 2019) and CIFAR-10v2 (Recht et al., 2018); (ii) **CIFAR-100** including the original dataset and CIFAR-100-C; (iii) all four BREEDs datasets (Santurkar et al., 2021), i.e., **Entity13**, **Entity30**, **Nonliving26**, **Living17**. BREEDs leverages class hierarchy in ImageNet (Russakovsky et al., 2015) to repurpose original classes to be the subpopulations and define a classification task on superclasses. We consider subpopulation shift and natural shifts induced due to differences in the data collection process of ImageNet, i.e, ImageNetv2 (Recht et al., 2019b) and a combination of both. (iv) **OfficeHome** (Venkateswara et al., 2017) which includes four domains: art, clipart, product, and real; (v) **DomainNet** (Peng et al., 2019) where we consider four domains: clipart, painting, real, sketch; (vi) **Visda** (Peng et al., 2017; 2018) which contains three domains: train, val and test; (vii) **FMoW** (Christie et al., 2018; Koh et al., 2021) from WILDS benchmark which includes three domains: train, OOD val, and OOD test—with satellite images taken in different geographical regions and at different times; (viii) **Camelyon** (Bandi et al., 2018) from WILDS benchmark which includes three domains: train, OOD val, and OOD test, for tumor identification with domains corresponding to different hospitals; (ix) **Civilcomments** (Borkan et al., 2019) which includes three domains: train, OOD val, and OOD test, for toxicity detection with domains corresponding to different demographic subpopulations; (x) **Retiring Adults** (Ding et al., 2021) where we consider the ACSIncome prediction task with various domains representing different states and time-period; and (xi) **Mimic Readmission** (Johnson et al., 2020; PhysioBank, 2000) where the task is to predict readmission risk with various domains representing data from different time-period.

Simulating a shift in target marginal The above datasets present minor to no shift in label marginal. Hence, we simulate such a shift by altering the target label marginal and keeping the source target distribution fixed (to the original source label distribution). Note that, unlike some previous studies, we do not alter the source label marginal because, in practice, we may have an option to carefully curate the training distribution but might have little to no control over the target label marginal.

For each target dataset, we have the true labels which allow us to vary the target label distribution. In particular, we sample the target label marginal from a Dirichlet distribution with a parameter $\alpha \in \{0.5, 1, 3.0, 10\}$ multiplier to the original target marginal. Specifically, $p_t(y) \sim \mathrm{Dir}(\beta)$ where $\beta_y = \alpha \cdot p_{t,0}(y)$ and $p_{t,0}(y)$ is the original target label marginal. The Dirichlet parameter α controls the severity of shift in target label marginal. Intuitively, as α decreases, the severity of the shift increases. For completeness, we also include the target dataset with the original target label marginal. For ease of exposition, we denote the shifts as None (no external shift) in the set of Dirichlet parameters, i.e. the limiting distribution as $\alpha \to \infty$. After simulating the shift in the target label marginal (with two seeds for each α), we obtain 560 pairs of different source and target datasets.

7.3.2 Domain Adaptation Methods

We implement the following algorithms (a more detailed description of each method is included in App. F.11):

Source only As a baseline, we include model trained with empirical risk minimization (Vapnik, 1999) with cross-entropy loss on the source domain. We include source only models trained with and without augmentations. We also include adversarial robust models trained on source data with augmentations (**Source (adv)**). In particular, we use models adversarially trained against ℓ_2-perturbations.

Domain alignment methods These methods employ domain-adversarial training schemes aimed to learn invariant representations across different domains (Ganin et al., 2016; Tan et al., 2020; Zhang et al., 2019). For our experiments, we include the following *five* methods: Domain Adversarial Neural Networks (**DANN** (Ganin et al., 2016)), Conditional DANN (**CDANN** (Long et al., 2018), Maximum Classifier Discrepancy (**MCD** (Saito et al., 2018a)), Importance-reweighted DANN and CDANN (i.e. **IW-DANN** & **IW-CDANN** Tachet des Combes et al. (2020)).

Self-training methods These methods "pseudo-label" unlabeled examples with the model's own predictions and then train on them as if they were labeled examples. For vision

Algorithm 9 Meta algorithm to handle label marginal shift

```
input Source training and validation data: $(X_S, Y_S)$ and $(X'_S, Y'_S)$, unlabeled target training
    and validation data: $X_T$ and $X'_T$, classifier $f$, and DA algorithm $\mathcal{A}$
$\widetilde{X}_S, \widetilde{Y}_S \leftarrow \text{SampleClassBalanced}(X_S, Y_S)$
                                                        {Balance source data}
for $t = 1$ to $T$ do
    $\widehat{w}Y_T \leftarrow \arg\max_y f_y(X_T)$
    $\widetilde{X}_T \leftarrow \text{SampleClassBalanced}(X_T, \widehat{w}Y_T)$
                                                        {Pseudo-balance target data}
    Run an epoch of $\mathcal{A}$ to update $f$ on balanced source data $\{\widetilde{X}_S, \widetilde{Y}_S\}$ and target data
    $\{\widetilde{X}_T\}$
end for
$\widehat{w}p_t(y) \leftarrow \text{EstimateLabelMarginal}(f, X'_S, Y'_S, X'_T)$
$f'_j \leftarrow \frac{\widehat{w}p_t(y=j) \cdot f_j}{\sum_k \widehat{w}p_t(y=k) \cdot f_k}$ for all $j \in \mathcal{Y}$
                                                        {Re-weight classifier}
output Target label marginal $\widehat{w}p_t(y)$ and classifier $f'$
```

datasets, these methods often also use consistency regularization, which encourages the model to make consistent predictions on augmented views of unlabeled examples (Berthelot et al., 2021; Lee et al., 2013; Xie et al., 2020b). We include the following three algorithms: **FixMatch** (Sohn et al., 2020), **Noisy Student** (Xie et al., 2020a), Selective Entropy Optimization via Committee Consistency (**SENTRY** (Prabhu et al., 2021)). For NLP and tabular dataset, where we do not have strong augmentations defined, we consider **PseudoLabel** algorithm (Lee et al., 2013).

Test-time adaptation methods These methods take a source model and adapt a few parameters (e.g. batch norm parameters, etc.) on the unlabeled target data with an aim to improve target performance. We include: **CORAL** (Sun et al., 2016) or Domain Adjusted Regression (DARE (Rosenfeld et al., 2022)), BatchNorm adaptation (**BN-adapt** (Li et al., 2016; Schneider et al., 2020)), Test entropy minimization (**TENT** (Wang et al., 2021a)).

7.3.3 Meta algorithm to handle target label marginal shift

Here we discuss two simple general-purpose corrections that we implement in our framework. First, note that, as the severity of shift in the target label marginal increases, the performance of DA methods can falter as the training is done over source and target datasets with different class proportions. Indeed, failure of domain adversarial training methods (one category of deep DA methods) has been theoretically and empirically shown in the literature (Wu et al., 2019; Zhao et al., 2019). In our experiments, we show that a failure due to a shift in label distribution is not limited to domain adversarial training methods, but is common with all the popular DA methods (Sec. 7.4).

Re-sampling To handle label imbalance in standard supervised learning, re-sampling the data to balance the class marginal is a known successful strategy (Buda et al., 2018; Cao et al., 2019b; Chawla et al., 2002). In relaxed label shift, we seek to handle the imbalance in the target data (with respect to the source label marginal), where we do not have access to true labels. We adopt an alternative strategy of leveraging pseudolabels for target data to perform pseudo class-balanced re-sampling[1] (Wei et al., 2021; Zou et al., 2018). For relaxed label shift problems, (Prabhu et al., 2021) employed this technique with their committee consistency objective, SENTRY. However, they did not explore re-sampling based correction for existing DA techniques. Since this technique can be used in conjunction with any DA methods, we employ this re-sampling technique with existing DA methods and find that re-sampling benefits all DA methods, often improving over SENTRY in our testbed (Sec. 7.4).

Re-weighting With re-sampling, we can hope to train the classifier $\widehat{w}f$ on a mixture of balanced source and balanced target datasets in an ideal case. However, this still leaves open the problem of adapting the classifier $\widehat{w}f$ to the original target label distribution which is not available. If we can estimate the target label marginal, we can post-hoc adapt the classifier $\widehat{w}f$ with a simple re-weighting correction (Alexandari et al., 2021; Lipton et al., 2018b). To estimate the target label marginal, we turn to techniques developed under the stricter label shift assumption (recall, the setting where $p(x|y)$ remains domain invariant). These approaches leverage off-the-shelf classifiers to estimate target marginal and provide $\mathcal{O}(1/\sqrt{n})$ convergence rates under the label shift condition with mild assumptions on the classifier (Azizzadenesheli et al., 2019; Garg et al., 2020a; Lipton et al., 2018b).

While the relaxed label shift scenario violates the conditions required for consistency of label shift estimation techniques, we nonetheless employ these techniques and empirically evaluate efficacy of these methods in our testbed. In particular, to estimate the target label marginal, we experiment with: (i) RLLS (Azizzadenesheli et al., 2019); (ii) MLLS (Alexandari et al., 2021); and (iii) *baseline estimator* that simply averages the prediction of a classifier f on unlabeled target data. We provide precise details about these methods in App. F.6. Since these methods leverage off-the-shelf classifiers, classifiers obtained with any DA methods can be used in conjunction with these estimation methods.

Summary Overall, in Algorithm 9, we illustrate how to incorporate the re-sampling and re-weighting correction with existing DA techniques. Fig. F.5 in App. F.5 illustrates the method. Algorithm $\mathcal{A}$ can be any DA method and in Step 7, we can use any of the three methods listed above to estimate the target label marginal. We instantiate Algorithm 9 with several algorithms from Sec. 7.3.2 in App. F.11. Intuitively, in an ideal scenario when the re-sampling step in our meta-algorithm perfectly corrects for label imbalance between source and target, we expect DA methods to adapt classifier f to $p(x|y)$ shift. The re-weighting step in our meta-algorithm can then adapt the classifier f to the target label marginal $p_t(y)$. We emphasize that in our work, we *do not* claim to propose these corrections. But, to the best of our knowledge, our work is the first to combine these two

[1] A different strategy could be to re-sample target pseudolabel marginal to match source label marginal. For simplicity, we choose to balance source label marginal and target pseudolabel marginal.

corrections together and perform extensive experiments across diverse datasets.

7.3.4 Other choices for realistic evaluation

For a fair evaluation and comparison across different datasets and DA algorithms, we re-implemented all the algorithms with consistent design choices whenever applicable. We also make several additional implementation choices, described below. We defer the additional details to App. F.12.

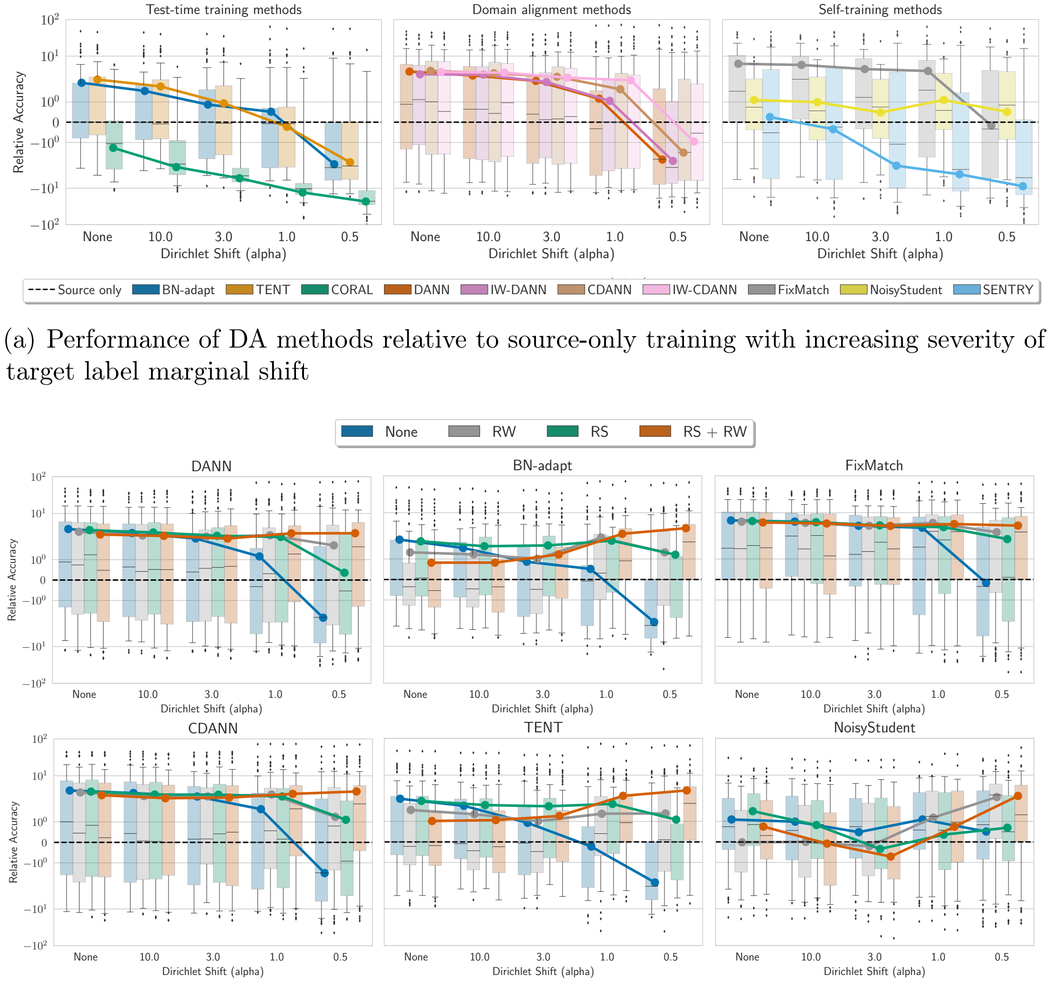
Model selection criteria and hyperparameters Given that we lack validation i.i.d data from the target distribution, model selection in DA problems *can not* follow the standard workflow used in supervised training. Prior works often omit details on how to choose hyperparameters leaving open a possibility of choosing hyperparameters using the test set which can provide a false and unreliable sense of improvement. Moreover, inconsistent hyperparameter selection strategies can complicate fair evaluations mis-associating the improvements to the algorithm under study.

In our work, we use source hold-out performance to pick the best hyperparameters. First, for ℓ_2 regularization and learning rate, we perform a sweep over random hyperparameters to maximize the performance of source only model on the hold-out source data. Then for each dataset, we keep these hyperparameters fixed across DA algorithms. For DA methods specific hyperparameters, we use the same hyperparameters across all the methods incorporating the suggestions made in corresponding papers. Within a run, we use hold out performance on the source to pick the early stopping point. In appendices, we report *oracle* performance by choosing the early stopping point with target accuracy.

Evaluation criteria To evaluate the target label marginal estimation, we report ℓ_1 error between the estimated label distribution and true label distribution. To evaluate the classifier performance on target data, we report performance of the (adapted) classifier on a hold-out partition of target data.

Architectural and pretraining details We experiment with different architectures (e.g., DenseNet121, Resenet18, Resnet50, DistilBERT, MLP and Transformer). We experiment with randomly-initialized models and Imagenet, and DistillBert pre-trained models. Given a dataset, we use the same architecture across different DA algorithms.

Data augmentation Data augmentation is a standard ingredient to train vision models which can approximate some of the variations between domains. Unless stated otherwise, we train all the vision datasets using the standard strong augmentation technique: random horizontal flips, random crops, augmentation with Cutout (DeVries and Taylor, 2017), and RandAugment (Cubuk et al., 2020). To understand help with data augmentations alone, we also experiment with source-only models trained without any data augmentation. For tabular and NLP datasets, we do not use any augmentations.



(a) Performance of DA methods relative to source-only training with increasing severity of target label marginal shift

(b) Performance of DA methods relative to source-only training when paired with our meta-algorithm (RS and RW corrections)

Figure 7.2: *Performance of different DA methods relative to a source-only model across all distribution shift pairs in vision datasets grouped by shift severity in label marginal.* We plot the relative accuracy of the model trained with that DA method by subtracting the accuracy of the source-only model. Smaller the Dirichlet shift parameter, the more severe is the shift in target class proportion. **(a)** Shifts with $\alpha = \{\text{None}, 10.0, 3.0\}$ have little to no impact on different DA methods whereas the performance of all DA methods degrades when $\alpha \in \{1.0, 0.5\}$ often falling below the performance of a source-only classifier (except for Noisy Student). **(b)** RS and RW (in our meta-algorithm) together significantly improve aggregate performance over no correction for all DA methods. While RS consistently helps (over no correction) across different label marginal shift severities, RW hurts slightly for BN-adapt, TENT, and NoisyStudent when shift severity is small. However, for severe shifts ($\alpha \in \{3.0, 1.0, 0.5\}$) RW significantly improves performance for all the methods. Parallel results on tabular and language datasets in App. F.2.

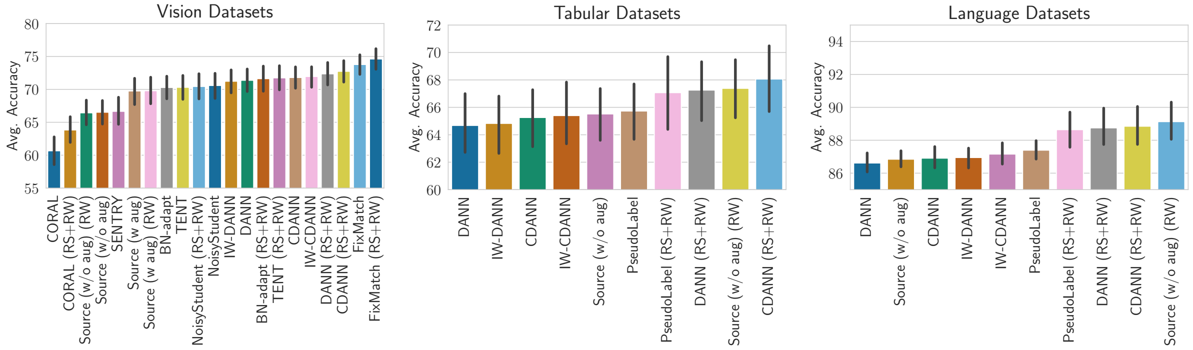


Figure 7.3: *Average accuracy of different DA methods aggregated across all distribution pairs in each modality.*

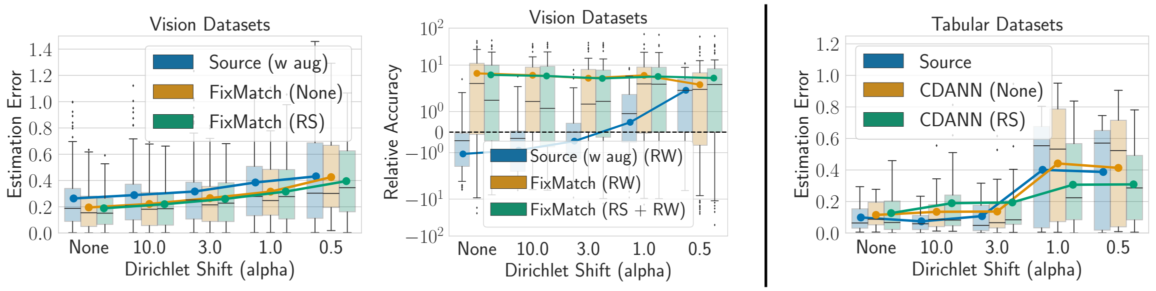


Figure 7.4: *Target label marginal estimation (ℓ_1) error and accuracy with RLLS and classifiers obtained with different DA methods.* **(Left)** Across all shift severities in vision datasets, RLLS with classifiers obtained with DA methods improves over RLLS with a source-only classifier. **(Right)** For tabular datasets, RLLS with classifiers obtained with DA methods improves over RLLS with a source-only classifier for severe target label marginal shifts. Plots for each DA method and all datasets are in App. F.8.

7.4 Main Results

We present aggregated results on vision datasets in our testbed in Fig. 7.2. In App. F.2, we present aggregated results on NLP and tabular datasets. Note that we do not include RS results with a source only model as it is trained only on source data and we observed no differences with just balancing the source data (as for most datasets source is already balanced) in our experiments. Unless specified otherwise, we use source validation performance as the early stopping criterion. Based on running our entire RLSBench suite, we distill our findings into the following takeaways.

Popular deep DA methods without any correction falter. While DA methods often improve over a source-only classifier for cases when the target label marginal shift is absent or low, the performance of these methods (except Noisy Student) drops below the performance of a source-only classifier when the shift in target label marginal is severe (i.e., when $\alpha = 0.5$ in Fig. 7.2a, F.1a, and F.2a). On the other hand, DA methods when paired with RS and RW correction, significantly improve over a source-only model even when the shift in target label marginal is severe (Fig. 7.2b, F.1b, and F.2b).

Re-sampling to pseudobalance target often helps all DA methods across all modalities. When the shift in target label marginal is absent or very small (i.e., $\alpha \in \{\textsc{None}, 10.0\}$ in Fig. 7.2b, F.1b, and F.2b), we observe no (significant) differences in performance with re-sampling. However, as the shift severity in target label marginal increases (i.e., $\alpha \in \{3.0, 1.0, 0.5\}$ in Fig. 7.2b, F.1b, and F.2b), we observe that re-sampling typically improves all DA methods in our testbed.

Benefits of post-hoc re-weighting of the classifier depends on shift severity and the underlying DA algorithm. For domain alignment methods (i.e. DANN and CDANN) and self-training methods, in particular FixMatch and PseudoLabel, we observe that RW correction typically improves (over no correction) significantly when the target label marginal shift is severe (i.e., $\alpha \in \{3.0, 1.0, 0.5\}$ in Fig. 7.2b, F.1b, and F.2b) and has no (significant) effect when the shift in target label marginal is absent or very small (i.e., $\alpha \in \{\textsc{None}, 10.0\}$ in Fig. 7.2b, F.1b, and F.2b). For BN-adapt, TENT, and NoisyStudent, RW correction can slightly hurt when target label marginal shift is absent or low (i.e., $\alpha \in \{\textsc{None}, 10.0\}$ in Fig. 7.2b) but continues to improve significantly when the target label marginal shift is severe (i.e., $\alpha \in \{3.0, 1.0, 0.5\}$ in Fig. 7.2b). Additionally, we observe that in specific scenarios of the real-world shift in $p(x|y)$ (e.g., subpopulation shift in BREEDs datasets, camelyon shifts, and replication study in CIFAR-10 which are benign relative to other vision dataset shifts in our testbed), RW correction does no harm to performance for BN-adapt, TENT, and NoisyStudent even when the target label marginal shift is less severe or absent.

DA methods paired with our meta-algorithm often improve over source-only classifier but no one method consistently performs the best. First, we observe that our source-only numbers are better than previously published results. Similar to previous studies (Gulrajani and Lopez-Paz, 2020), this can be attributed to improved design choices (e.g. data augmentation, hyperparameters) which we make consistent across all methods. While there is no consistent method that does the best across datasets, overall, FixMatch with RS and RW (our meta-algorithm) performs the best for vision datasets. For NLP datasets, source-only with RW (our meta-algorithm) performs the best overall. For tabular datasets, CDANN with RS and RW (our meta-algorithm) performs the best overall (Fig. 7.3).

Existing DA methods when paired with our meta-algorithm significantly outperform other DA methods specifically proposed for relaxed label shift. We observe that, with consistent experimental design across different methods, existing DA methods with RS and RW corrections often improve over previously proposed methods specifically aimed to tackle relaxed label shift, i.e., IW-CDANN, IW-DANN, and SENTRY (Fig. F.3). For severe target label marginal shifts, the performance of IW-DANN, IW-CDANN, and SENTRY often falls below that of the source-only model. Moreover, while the importance weighting (i.e., IW-CDANN and IW-DANN) improves over CDANN and DANN resp. (Fig. 7.2a, F.1a and F.2a), RS and RW corrections significantly outweigh those improvements (Fig. F.3).

BN-adapt and TENT with our meta-algorithm are simple and strong base-

lines. For models with batch norm parameters, BN-adapt (and TENT) with RS and RW steps is a computationally efficient and strong baseline. We observe that while the performance of BN-adapt (and TENT) can drop substantially when the target label marginal shifts (i.e., $\alpha \in \{1.0, 0.5\}$ in Fig. 7.2(a)), RS and RW correction improves the performance often improving BN-adapt (and TENT) over all other DA methods when the shift in target label marginal is extreme (i.e., $\alpha = 0.5$ in Fig. 7.2(b)).

DA methods yield better target label marginal estimates, and hence larger accuracy improvements with re-weighting, than source-only classifiers. Recall that we experiment with target label marginal estimation methods that leverage off-the-shelf classifiers to obtain an estimate. We observe that estimators leveraging DA classifiers tend to perform better than using source-only classifiers for tabular and vision datasets (Fig. 7.4). For NLP, we observe that DA classifier and source-only classifier have performance (with source-only often performing slightly better). Correspondingly, as one might expect, better estimation yields greater accuracy improvements when applying our RW correction. In particular, RW correction with DA methods improves over the source-only classifier for vision and tabular datasets and vice-versa for NLP datasets. (Fig. 7.4).

Early stopping criterion matters. We observe a consistent $\approx$2% and $\approx$8% accuracy difference on vision and tabular datasets respectively with all methods (Fig. F.16). On NLP datasets, while the early stopping criteria have $\approx$2% accuracy difference when RW and RS corrections are not employed, the difference becomes negligible when these corrections are employed (Fig. F.16). These results highlight that subsequent works should describe the early stopping criteria used within their evaluations.

Data augmentation helps. Corroborating findings from previous studies in other settings (Gulrajani and Lopez-Paz, 2020; Sagawa et al., 2021), we observe that data augmentation can improve the performance of a source-only model on vision datasets in relaxed label shift scenarios. Thus, whenever applicable, subsequent methods should use data augmentations.

7.5 Conclusion

Our work is the first large-scale study investigating methods under the relaxed label shift scenario. Relative to works operating strictly under the label shift assumption, RLSBench provides an opportunity for sensitivity analysis, allowing researchers to measure the robustness of their methods under various sorts of perturbations to the class-conditional distributions. Relative to the benchmark-driven deep domain adaptation literature, our work provides a comprehensive and standardized suite for evaluating under shifts in label distributions, bringing these benchmarks one step closer to exhibit the sort of diversity that we should expect to encounter when deploying models in the wild. On one hand, the consistent improvements observed from label shift adjustments are promising. At the same time, given the underspecified nature of the problem, practitioners must remain vigilant and take performance on any benchmark with a grain of salt, considering the various ways that it might (or might not) be representative of the sorts of situations that might arise in their application of interest.

Also, we observe that the success of target label marginal estimation techniques depends on the nature of the shifts in $p(x|y)$. In follow up work (Kannan et al., 2024), we mathematically characterize the behavior of label shift estimation techniques when the label shift assumption is violated.

Part III

Evaluating Models Without Access to Labeled Data

Chapter 8

RATT: Leveraging Unlabeled Data to Guarantee Generalization

Based on Garg et al. (2021a): Garg, Saurabh, Sivaraman Balakrishnan, Zico Kolter, and Zachary Lipton. Ratt: Leveraging unlabeled data to guarantee generalization. In International Conference on Machine Learning, 2021

Abstract

To assess generalization, machine learning scientists typically either (i) bound the generalization gap and then (after training) plug in the empirical risk to obtain a bound on the true risk; or (ii) validate empirically on holdout data. However, (i) typically yields vacuous guarantees for overparameterized models; and (ii) shrinks the training set and its guarantee erodes with each re-use of the holdout set. In this chapter, we leverage unlabeled data to produce generalization bounds. After augmenting our (labeled) training set with randomly labeled data, we train in the standard fashion. Whenever classifiers achieve low error on the clean data but high error on the random data, our bound ensures that the true risk is low. We prove that our bound is valid for 0-1 empirical risk minimization and with linear classifiers trained by gradient descent. Our approach is especially useful in conjunction with deep learning due to the early learning phenomenon whereby networks fit true labels before noisy labels but requires one intuitive assumption. Empirically, on canonical computer vision and NLP tasks, our bound provides non-vacuous generalization guarantees that track actual performance closely. This work enables practitioners to certify generalization even when (labeled) holdout data is unavailable and provides insights into the relationship between random label noise and generalization. Code is available at this url.

8.1 Introduction

Addressing model adaptation in the context of unlabeled data under distribution shift is a critical challenge. Equally crucial is the evaluation of these adapted models to gauge the impact of distribution shift. In this section of the thesis, we delve into the evaluation of models using only unlabeled data from the target distribution of interest. Specifically, this chapter will concentrate on the fundamental scenario where there is no distribution shift, aiming to assess model generalization in the absence of labeled holdout data.

Typically, machine learning scientists establish generalization in one of two ways. One approach, favored by learning theorists, places an *a priori* bound on the gap between the empirical and true risks, usually in terms of the complexity of the hypothesis class. After fitting the model on the available data, one can plug in the empirical risk to obtain a guarantee on the true risk. The second approach, favored by practitioners, involves splitting the available data into training and holdout partitions, fitting the models on the former and estimating the population risk with the latter.

Surely, both approaches are useful, with the former providing theoretical insights and the latter guiding the development of a vast array of practical technology. Nevertheless, both methods have drawbacks. Most *a priori* generalization bounds rely on uniform convergence and thus fail to explain the ability of overparameterized networks to generalize (Nagarajan and Kolter, 2019b; Zhang et al., 2017). On the other hand, provisioning a holdout dataset restricts the amount of labeled data available for training. Moreover, risk estimates based on holdout sets lose their validity with successive re-use of the holdout data due to adaptive overfitting (Blum and Hardt, 2015; Dwork et al., 2015; Murphy, 2012). However, recent empirical studies suggest that on large benchmark datasets, adaptive overfitting is surprisingly absent (Recht et al., 2019b).

In this chapter, we propose Randomly Assign, Train and Track (RATT), a new method that leverages unlabeled data to provide a *post-training* bound on the true risk (i.e., the population error). Here, we assign random labels to a fresh batch of unlabeled data, augmenting the clean training dataset with these randomly labeled points. Next, we train on this data, following standard risk minimization practices. Finally, we track the error on the randomly labeled portion of training data, estimating the error on the mislabeled portion and using this quantity to upper bound the population error.

Counterintuitively, we guarantee generalization by guaranteeing overfitting. Specifically, we prove that Empirical Risk Minimization (ERM) with 0-1 loss leads to lower error on the *mislabeled training data* than on the *mislabeled population*. Thus, if despite minimizing the loss on the combined training data, we nevertheless have high error on the mislabeled portion, then the (mislabeled) population error will be even higher. Then, by complementarity, the (clean) population error must be low. Finally, we show how to obtain this guarantee using randomly labeled (vs mislabeled data), thus enabling us to incorporate unlabeled data.

To expand the applicability of our idea beyond ERM on 0-1 error, we prove corresponding

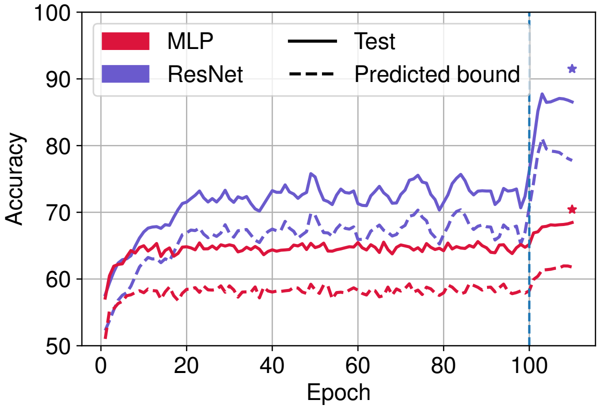


Figure 8.1: **Predicted lower bound on the clean population error** with ResNet and MLP on binary CIFAR. Results aggregated over 5 seeds. '*' denotes the best test performance achieved when training with only clean data and the same hyperparameters (except for the stopping point). The bound predicted by RATT (RHS in (8.2)) closely tracks the population accuracy on clean data.

results for a linear classifier trained by gradient descent to minimize squared loss. Furthermore, leveraging the connection between early stopping and ℓ_2-regularization in linear models (Ali et al., 2018; 2020; Suggala et al., 2018), our results extend to early-stopped gradient descent. Because we make no assumptions on the data distribution, our results on linear models hold for more complex models such as kernel regression and neural networks in the Neural Tangent Kernel (NTK) regime (Allen-Zhu et al., 2019b; Chizat et al., 2019; Du et al., 2019; 2018; Jacot et al., 2018).

Addressing practical deep learning models, our guarantee requires an additional (reasonable) assumption. Our experiments show that the bound yields non-vacuous guarantees that track test error across several major architectures on a range of benchmark datasets for computer vision and Natural Language Processing (NLP). Because, in practice, overparameterized deep networks exhibit an *early learning phenomenon*, fitting clean data before mislabeled data (Arora et al., 2019a; Li et al., 2019; Liu et al., 2020), our procedure yields tight bounds in the early phases of learning. Experimentally, we confirm the early learning phenomenon in standard Stochastic Gradient Descent (SGD) training and illustrate the effectiveness of weight decay combined with large initial learning rates in avoiding interpolation to mislabeled data while maintaining fit on the training data, strengthening the guarantee provided by our method.

To be clear, we do not advocate RATT as a blanket replacement for the holdout approach. Our main contribution is to introduce a new theoretical perspective on generalization and to provide a method that may be applicable even when the holdout approach is unavailable. Of interest, unlike generalization bounds based on uniform-convergence that restrict the complexity of the hypothesis class (Bartlett et al., 2017; Nagarajan and Kolter, 2019a; Neyshabur et al., 2015; 2017b; 2018), our *post hoc* bounds depend only on the fit to mislabeled data. We emphasize that our theory does not guarantee *a priori* that early learning should take place but only *a posteriori* that when it does, we can provide non-vacuous bounds on the population error. Conceptually, this finding underscores the significance of the early learning phenomenon in the presence of noisy labels and motivates further work to explain why it occurs.

8.2 Preliminaries

By $\|\cdot\|$, and $\langle\cdot,\cdot\rangle$ we denote the Euclidean norm and inner product, respectively. For a vector $v \in \mathbb{R}^d$, we use v_j to denote its j^{th} entry, and for an event E we let $\mathbb{I}\left[E\right]$ denote the binary indicator of the event.

Suppose we have a multiclass classification problem with the input domain $\mathcal{X} \subseteq \mathbb{R}^d$ and label space $\mathcal{Y} = \{1, 2, \ldots, k\}$[1]. By $\mathcal{D}$, we denote the distribution over $\mathcal{X} \times \mathcal{Y}$. A dataset $S := \{(x_i, y_i)\}_{i=1}^n \sim \mathcal{D}^n$ contains n points sampled i.i.d. from $\mathcal{D}$. By $\mathcal{S}$, $\mathcal{T}$, and $\widetilde{\mathcal{S}}$, we denote the (uniform) empirical distribution over points in datasets S, T, and $\widetilde{S}$, respectively. Let $\mathcal{F}$ be a class of hypotheses mapping $\mathcal{X}$ to $\mathbb{R}^k$. A *training algorithm* $\mathcal{A}$: takes a dataset S and returns a classifier $f(\mathcal{A}, S) \in \mathcal{F}$. When the context is clear, we drop the parentheses for convenience. Given a classifier f and datum (x, y), we denote the 0-1 error (i.e., classification error) on that point by $\mathcal{E}(f(x), y) := \mathbb{I}\left[y \notin \arg\max_{j\in\mathcal{Y}} f_j(x)\right]$, We express the *population error* on $\mathcal{D}$ as $\mathcal{E}_{\mathcal{D}}(f) := \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathcal{E}(f(x), y)\right]$ and the *empirical error* on S as $\mathcal{E}_{\mathcal{S}}(f) := \mathbb{E}_{(x,y)\sim\mathcal{S}}\left[\mathcal{E}(f(x), y)\right] = \frac{1}{n}\sum_{i=1}^n \mathcal{E}(f(x_i), y_i)$.

Throughout, we consider a *random label assignment* procedure: draw $x \sim \mathcal{D}_{\mathcal{X}}$ (the underlying distribution over $\mathcal{X}$), and then assign a label sampled uniformly at random. We denote a randomly labeled dataset by $\widetilde{S} := \{(x_i, y_i)\}_{i=1}^m \sim \widetilde{\mathcal{D}}^m$, where $\widetilde{\mathcal{D}}$ is the distribution of randomly labeled data. By $\mathcal{D}'$, we denote the mislabeled distribution that corresponds to selecting examples (x, y) according to $\mathcal{D}$ and then re-assigning the label by sampling among the incorrect labels $y' \neq y$ (renormalizing the label marginal).

8.3 Generalization Bound for RATT with ERM

We now present our generalization bound and proof sketches for ERM on the 0-1 loss (full proofs in App. G.1). For any dataset T, ERM returns the classifier $\widehat{f}$ that minimizes the empirical error:

$$\widehat{f} := \operatorname*{arg\,min}_{f\in\mathcal{F}} \mathcal{E}_{\mathcal{T}}(f)\,. \tag{8.1}$$

We focus first on binary classification. Assume we have a clean dataset $S \sim \mathcal{D}^n$ of n points and a randomly labeled dataset $\widetilde{S} \sim \widetilde{\mathcal{D}}^m$ of m $(< n)$ points with labels in $\widetilde{S}$ are assigned uniformly at random. We show that with 0-1 loss minimization on the union of S and $\widetilde{S}$, we obtain a classifier whose error on $\mathcal{D}$ is upper bounded by a function of the empirical errors on clean data $\mathcal{E}_{\mathcal{S}}$ (lower is better) and on randomly labeled data $\mathcal{E}_{\widetilde{\mathcal{S}}}$ (higher is better):

[1] For binary classification, we use $\mathcal{Y} = \{-1, 1\}$.

Theorem 8.3.1. *For any classifier $\widehat{w}f$ obtained by ERM* (8.1) *on dataset $S \cup \widetilde{S}$, for any $\delta > 0$, with probability at least $1 - \delta$, we have*

$$\begin{aligned}\mathcal{E}_{\mathcal{D}}(\widehat{f}) \leqslant\, & \mathcal{E}_{\mathcal{S}}(\widehat{f}) + 1 - 2\mathcal{E}_{\widetilde{\mathcal{S}}}(\widehat{f}) \\ & + \left(\sqrt{2}\mathcal{E}_{\widetilde{\mathcal{S}}}(\widehat{f}) + 2 + \frac{m}{2n}\right)\sqrt{\frac{\log(4/\delta)}{m}}\,. \end{aligned} \tag{8.2}$$

In short, this theorem tells us that if after training on both clean and randomly labeled data, we achieve low error on the clean data but high error (close to $1/2$) on the randomly labeled data, then low population error is guaranteed. Note that because the labels in $\widetilde{S}$ are assigned randomly, the error $\mathcal{E}_{\widetilde{\mathcal{S}}}(f)$ for any fixed predictor f (not dependent on $\widetilde{S}$) will be approximately $1/2$. Thus, if ERM produces a classifier that has not fit to the randomly labeled data, then $(1 - 2\mathcal{E}_{\widetilde{\mathcal{S}}}(\widehat{f}))$ will be approximately 0, and our error will be determined by the fit to clean data. The final term accounts for finite sample error—notably, it (i) does not depend on the complexity of the hypothesis class; and (ii) approaches 0 at a $\mathcal{O}(1/\sqrt{m})$ rate (for $m < n$).

Our proof strategy unfolds in three steps. First, in Lemma 8.3.2 we bound $\mathcal{E}_{\mathcal{D}}(\widehat{w}f)$ in terms of the error on the mislabeled subset of $\widetilde{S}$. Next, in Lemmas 8.3.3 and 8.3.4, we show that the error on the mislabeled subset can be accurately estimated using only clean and randomly labeled data.

To begin, assume that we actually knew the original labels for the randomly labeled data. By $\widetilde{S}_C$ and $\widetilde{S}_M$, we denote the clean and mislabeled portions of the randomly labeled data, respectively (with $\widetilde{S} = \widetilde{S}_M \cup \widetilde{S}_C$). Note that for binary classification, a lower bound on mislabeled population error $\mathcal{E}_{\mathcal{D}'}(\widehat{w}f)$ directly upper bounds the error on the original population $\mathcal{E}_{\mathcal{D}}(\widehat{w}f)$. Thus we only need to prove that the empirical error on the mislabeled portion of our data is lower than the error on unseen mislabeled data, i.e., $\mathcal{E}_{\widetilde{\mathcal{S}}_M}(\widehat{w}f) \leqslant \mathcal{E}_{\mathcal{D}'}(\widehat{w}f) = 1 - \mathcal{E}_{\widetilde{\mathcal{S}}_M}(\widehat{f})$ (upto $\mathcal{O}(1/\sqrt{m})$).

Lemma 8.3.2. *Assume the same setup as in Theorem 8.3.1. Then for any $\delta > 0$, with probability at least $1 - \delta$ over the random draws of mislabeled data $\widetilde{S}_M$, we have*

$$\mathcal{E}_{\mathcal{D}}(\widehat{f}) \leqslant 1 - \mathcal{E}_{\widetilde{\mathcal{S}}_M}(\widehat{f}) + \sqrt{\frac{\log(1/\delta)}{m}}\,. \tag{8.3}$$

Proof Sketch. The main idea of our proof is to regard the clean portion of the data $(S \cup \widetilde{S}_C)$ as fixed. Then, there exists a classifier f^* that is optimal over draws of the mislabeled data $\widetilde{S}_M$. Formally,

$$f^* := \arg\min_{f \in \mathcal{F}} \mathcal{E}_{\breve{\mathcal{D}}}(f),$$

where $\breve{\mathcal{D}}$ is a combination of the *empirical distribution* over correctly labeled data $S \cup \widetilde{S}_C$ and the (population) distribution over mislabeled data $\mathcal{D}'$. Recall that $\widehat{f} := \arg\min_{f \in \mathcal{F}} \mathcal{E}_{\mathcal{S} \cup \widetilde{\mathcal{S}}}(f)$.

Since, $\widehat{f}$ minimizes 0-1 error on $S \cup \widetilde{S}$, we have $\mathcal{E}_{\mathcal{S}\cup\widetilde{\mathcal{S}}}(\widehat{f}) \leqslant \mathcal{E}_{\mathcal{S}\cup\widetilde{\mathcal{S}}}(f^*)$. Moreover, since f^* is independent of $\widetilde{S}_M$, we have with probability at least $1-\delta$ that

$$\mathcal{E}_{\widetilde{\mathcal{S}}_M}(f^*) \leqslant \mathcal{E}_{\mathcal{D}'}(f^*) + \sqrt{\frac{\log(1/\delta)}{m}}\,.$$

Finally, since f^* is the optimal classifier on $\breve{\mathcal{D}}$, we have $\mathcal{E}_{\breve{\mathcal{D}}}(f^*) \leqslant \mathcal{E}_{\breve{\mathcal{D}}}(\widehat{f})$. Combining the above steps and using the fact that $\mathcal{E}_{\mathcal{D}} = 1 - \mathcal{E}_{\mathcal{D}'}$, we obtain the desired result. □

While the LHS in (8.3) depends on the unknown portion $\widetilde{S}_M$, our goal is to use unlabeled data (with randomly assigned labels) for which the mislabeled portion cannot be readily identified. Fortunately, we do not need to identify the mislabeled points to estimate the error on these points in aggregate $\mathcal{E}_{\widetilde{\mathcal{S}}_M}(\widehat{f})$. Note that because the label marginal is uniform, approximately half of the data will be correctly labeled and the remaining half will be mislabeled. Consequently, we can utilize the value of $\mathcal{E}_{\widetilde{\mathcal{S}}}(\widehat{f})$ and an estimate of $\mathcal{E}_{\widetilde{\mathcal{S}}_C}(\widehat{f})$ to lower bound $\mathcal{E}_{\widetilde{\mathcal{S}}_M}(\widehat{f})$. We formalize this as follows:

Lemma 8.3.3. *Assume the same setup as Theorem 8.3.1. Then for any $\delta > 0$, with probability at least $1-\delta$ over the random draws of $\widetilde{S}$, we have* $\left|2\mathcal{E}_{\widetilde{\mathcal{S}}}(\widehat{f}) - \mathcal{E}_{\widetilde{\mathcal{S}}_C}(\widehat{f}) - \mathcal{E}_{\widetilde{\mathcal{S}}_M}(\widehat{f})\right| \leqslant 2\mathcal{E}_{\widetilde{\mathcal{S}}}(\widehat{f})\sqrt{\frac{\log(4/\delta)}{2m}}\,.$

To complete the argument, we show that due to the exchangeability of the clean data S and the clean portion of the randomly labeled data S_C, we can estimate the error on the latter $\mathcal{E}_{\widetilde{\mathcal{S}}_C}(\widehat{w}f)$ by the error on the former $\mathcal{E}_{\mathcal{S}}(\widehat{w}f)$.

Lemma 8.3.4. *Assume the same setup as Theorem 8.3.1. Then for any $\delta > 0$, with probability at least $1-\delta$ over the random draws of $\widetilde{S}_C$ and S, we have* $\left|\mathcal{E}_{\widetilde{\mathcal{S}}_C}(\widehat{f}) - \mathcal{E}_{\mathcal{S}}(\widehat{f})\right| \leqslant \left(1 + \frac{m}{2n}\right)\sqrt{\frac{\log(2/\delta)}{m}}\,.$

Lemma 8.3.4 establishes a tight bound on the difference of the error of classifier $\widehat{f}$ on $\widetilde{S}_C$ and on S. The proof uses Hoeffding's inequality for randomly sampled points from a fixed population (Bardenet et al., 2015; Hoeffding, 1994).

Having established these core components, we can now summarize the proof strategy for Theorem 8.3.1. We bound the population error on clean data (the term on the LHS of (8.2)) in three steps: (i) use Lemma 8.3.2 to upper bound the error on clean distribution $\mathcal{E}_{\mathcal{D}}(\widehat{f})$, by the error on mislabeled training data $\mathcal{E}_{\widetilde{\mathcal{S}}_M}(\widehat{f})$; (ii) approximate $\mathcal{E}_{\widetilde{\mathcal{S}}_M}(\widehat{f})$ by $\mathcal{E}_{\widetilde{\mathcal{S}}_C}(\widehat{f})$ and the error on randomly labeled training data (i.e., $\mathcal{E}_{\widetilde{\mathcal{S}}}(\widehat{f})$) using Lemma 8.3.3; and (iii) use Lemma 8.3.4 to estimate $\mathcal{E}_{\widetilde{\mathcal{S}}_C}(\widehat{f})$ using the error on clean training data ($\mathcal{E}_{\mathcal{S}}(\widehat{f})$).

Comparison with Rademacher bound Our bound in Theorem 8.3.1 shows that we can upper bound the clean population error of a classifier by estimating its accuracy on the clean and randomly labeled portions of the training data. Next, we show that our bound's dominating term is upper bounded by the *Rademacher complexity* (Shalev-Shwartz and Ben-David, 2014), a standard distribution-dependent complexity measure.

Proposition 8.3.5. *Fix a randomly labeled dataset $\widetilde{S} \sim \widetilde{\mathcal{D}}^m$. Then for any classifier $f \in \mathcal{F}$ (possibly dependent on $\widetilde{S}$)*[2] *and for any $\delta > 0$, with probability at least $1 - \delta$ over random draws of $\widetilde{S}$, we have*

$$1 - 2\mathcal{E}_{\widetilde{S}}(f) \leqslant \mathbb{E}_{\epsilon,x}\left[\sup_{f \in \mathcal{F}}\left(\frac{\sum_i \epsilon_i f(x_i)}{m}\right)\right] + \sqrt{\frac{2\log(\frac{2}{\delta})}{m}},$$

where ϵ is drawn from a uniform distribution over $\{-1, 1\}^m$ and x is drawn from $\mathcal{D}_{\mathcal{X}}^m$.

In other words, the proposition above highlights that the accuracy on the randomly labeled data is never larger than the Rademacher complexity of $\mathcal{F}$ w.r.t. the underlying distribution over $\mathcal{X}$, implying that our bound is never looser than a bound based on Rademacher complexity. The proof follows by application of the bounded difference condition and McDiarmid's inequality (McDiarmid, 1989). We now discuss extensions of Theorem 8.3.1 to regularized ERM and multiclass classification.

Extension to regularized ERM Consider any function $R : \mathcal{F} \to \mathbb{R}$, e.g., a regularizer that penalizes some measure of complexity for functions in class $\mathcal{F}$. Consider the following regularized ERM:

$$\widehat{f} := \arg\min_{f \in \mathcal{F}} \mathcal{E}_{\mathcal{S}}(f) + \lambda R(f)\,, \tag{8.4}$$

where λ is a regularization constant. If the regularization coefficient is independent of the training data $S \cup \widetilde{S}$, then our guarantee (Theorem 8.3.1) holds. Formally,

Theorem 8.3.6. *For any regularization function R, assume we perform regularized ERM as in (8.4) on $S \cup \widetilde{S}$ and obtain a classifier $\widehat{f}$. Then, for any $\delta > 0$, with probability at least $1 - \delta$, we have $\mathcal{E}_{\mathcal{D}}(\widehat{f}) \leqslant \mathcal{E}_{\mathcal{S}}(\widehat{f}) + 1 - 2\mathcal{E}_{\widetilde{S}}(\widehat{f}) + \left(\sqrt{2}\mathcal{E}_{\widetilde{S}}(\widehat{f}) + 2 + \frac{m}{2n}\right)\sqrt{\frac{\log(1/\delta)}{m}}$.*

A key insight here is that the proof of Theorem 8.3.1 treats the clean data S as fixed and considers random draws of the mislabeled portion. Thus a data-independent regularization function does not alter our chain of arguments and hence, has no impact on the resulting inequality. We prove this result formally in App. G.1.

We note one immediate corollary from Theorem 8.3.6: when learning any function f parameterized by w with L_2-norm penalty on the parameters w, the population error with $\widehat{f}$ is determined by the error on the clean training data as long as the error on randomly labeled data is high (close to 1/2).

Extension to multiclass classification Thus far, we have addressed binary classification. We now extend these results to the multiclass setting. As before, we obtain datasets S and $\widetilde{S}$. Here, random labels are assigned uniformly among all classes.

[2] We restrict $\mathcal{F}$ to functions which output a label in $\mathcal{Y} = \{-1, 1\}$.

Theorem 8.3.7. *For any regularization function R, assume we perform regularized ERM as in (8.4) on $S \cup \widetilde{S}$ and obtain a classifier $\widehat{f}$. For a multiclass classification problem with k classes, for any $\delta > 0$, with probability at least $1-\delta$, we have*

$$\mathcal{E}_{\mathcal{D}}(\widehat{f}) \leqslant \mathcal{E}_{\mathcal{S}}(\widehat{f}) + (k-1)\left(1 - \tfrac{k}{k-1}\mathcal{E}_{\widetilde{\mathcal{S}}}(\widehat{f})\right) + c\sqrt{\frac{\log(\frac{4}{\delta})}{2m}}\,, \tag{8.5}$$

for some constant $c \leqslant (2k + \sqrt{k} + \frac{m}{n\sqrt{k}})$.

We first discuss the implications of Theorem 8.3.7. Besides empirical error on clean data, the dominating term in the above expression is given by $(k-1)\left(1 - \frac{k}{k-1}\mathcal{E}_{\widetilde{\mathcal{S}}}(\widehat{f})\right)$. For any predictor f (not dependent on $\widetilde{S}$), the term $\mathcal{E}_{\widetilde{\mathcal{S}}}(\widehat{f})$ would be approximately $(k-1)/k$ and for $\widehat{f}$, the difference now evaluates to the accuracy of the randomly labeled data. Note that for binary classification, (8.5) simplifies to Theorem 8.3.1.

The core of our proof involves obtaining an inequality similar to (8.3). While for binary classification, we could upper bound $\mathcal{E}_{\widetilde{\mathcal{S}}_M}$ with $1 - \mathcal{E}_{\mathcal{D}}$ (in the proof of Lemma 8.3.2), for multiclass classification, error on the mislabeled data and accuracy on the clean data in the population are not so directly related. To establish an inequality analogous to (8.3), we break the error on the (unknown) mislabeled data into two parts: one term corresponds to predicting the true label on mislabeled data, and the other corresponds to predicting neither the true label nor the assigned (mis-)label. Finally, we relate these errors to their population counterparts to establish an inequality similar to (8.3).

8.4 Generalization Bound for RATT with Gradient Descent

In the previous section, we presented results with ERM on 0-1 loss. While minimizing the 0-1 loss is hard in general, these results provide important theoretical insights. In this section, we show parallel results for linear models trained with Gradient Descent (GD).

To begin, we introduce the setup and some additional notation. For simplicity, we begin discussion with binary classification with $\mathcal{X} = \mathbb{R}^d$. Define a linear function $f(x;w) := w^T x$ for some $w \in \mathbb{R}^d$ and $x \in \mathcal{X}$. Given training set S, we suppose that the parameters of the linear function are obtained via gradient descent on the following L_2 regularized problem:

$$\mathcal{L}_S(w;\lambda) := \sum_{i=1}^{n}(w^T x_i - y_i)^2 + \lambda \left\| w \right\| 2^2\,, \tag{8.6}$$

where $\lambda \geqslant 0$ is a regularization parameter. Our choice to analyze squared loss minimization for linear networks is motivated in part by its analytical convenience, and follows recent theoretical work which analyze neural networks trained via squared loss minimization in the Neural Tangent Kernel (NTK) regime when they are well approximated by linear networks (Arora et al., 2019a; Du et al., 2019; Hu et al., 2019; Jacot et al., 2018). Moreover, recent research suggests that for classification tasks, squared loss minimization performs comparably to cross-entropy loss minimization (Hui and Belkin, 2020; Muthukumar et al., 2020).

For a given training set S, we use $S_{(i)}$ to denote the training set S with the i^{th} point removed. We now introduce one stability condition:

Condition 8.4.1 (Hypothesis Stability)**.** *We have β hypothesis stability if our training algorithm $\mathcal{A}$ satisfies the following for all $i \in \{1, 2, \dots, n\}$:*

$$\mathbb{E}_{\mathcal{S},(x,y)\in\mathcal{D}}\left[\left|\mathcal{E}\left(f(x), y\right) - \mathcal{E}\left(f_{(i)}(x), y\right)\right|\right] \leqslant \frac{\beta}{n},$$

where $f_{(i)} := f(\mathcal{A}, S_{(i)})$ and $f := f(\mathcal{A}, S)$.

This condition is similar to a notion of stability called *hypothesis stability* (Bousquet and Elisseeff, 2002; Elisseeff et al., 2003; Kearns and Ron, 1999). Intuitively, Condition 8.4.1 states that empirical leave-one-out error and average population error of leave-one-out classifiers are close. This condition is mild and does not guarantee generalization. We discuss the implications in more detail in App. G.2.3.

Now we present the main result of this section. As before, we assume access to a clean dataset $S = \{(x_i, y_i)\}_{i=1}^{n} \sim \mathcal{D}^n$ and randomly labeled dataset $\widetilde{S} = \{(x_i, y_i)\}_{i=n+1}^{n+m} \sim \widetilde{\mathcal{D}}^m$. Let $\boldsymbol{X} = [x_1, x_2, \cdots, x_{m+n}]$ and $\boldsymbol{y} = [y_1, y_2, \cdots, y_{m+n}]$. Fix a positive learning rate η such that $\eta \leqslant 1/\left(\left\|\boldsymbol{X}^T\boldsymbol{X}\right\| \mathrm{op} + \lambda^2\right)$ and an initialization $w_0 = 0$. Consider the following gradient descent iterates to minimize objective (8.6) on $S \cup \widetilde{S}$:

$$w_t = w_{t-1} - \eta \nabla_w \mathcal{L}_{S\cup\widetilde{S}}(w_{t-1}; \lambda) \quad \forall t = 1, 2, \dots . \tag{8.7}$$

Then we have $\{w_t\}$ converge to the limiting solution $\widehat{w}w = \left(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y}$. Define $\widehat{f}(x) := f(x; \widehat{w}w)$.

Theorem 8.4.2. *Assume that this gradient descent algorithm satisfies Condition 8.4.1 with $\beta = \mathcal{O}(1)$. Then for any $\delta > 0$, with probability at least $1-\delta$ over the random draws of datasets $\widetilde{S}$ and S, we have:*

$$\begin{aligned}\mathcal{E}_{\mathcal{D}}(\widehat{f}) \leqslant \mathcal{E}_{\mathcal{S}}(\widehat{f}) + 1 - 2\mathcal{E}_{\widetilde{\mathcal{S}}}(\widehat{f}) + \sqrt{\frac{4}{\delta}\left(\frac{1}{m} + \frac{3\beta}{m+n}\right)} \\ + \left(\sqrt{2}\mathcal{E}_{\widetilde{\mathcal{S}}}(\widehat{f}) + 1 + \frac{m}{2n}\right)\sqrt{\frac{\log(4/\delta)}{m}}\,.\end{aligned} \tag{8.8}$$

With a mild regularity condition, we establish the same bound on GD training with squared loss, notably the same dominating term on the population error, as in Theorem 8.3.1. In App. G.2.2, we present the extension to multiclass classification, where we again obtain a result parallel to Theorem 8.3.7.

Proof Sketch. Because squared loss minimization does not imply 0-1 error minimization, we cannot use arguments from Lemma 8.3.2. This is the main technical difficulty. To compare the 0-1 error at a train point with an unseen point, we use the closed-form expression for $\widehat{w}$. We show that the train error on mislabeled points is less than the population error on the distribution of mislabeled data (parallel to Lemma 8.3.2).

For a mislabeled training point (x_i, y_i) in $\widetilde{S}$, we show that

$$\mathbb{I}\left[y_i x_i^T \widehat{w}w \leqslant 0\right] \leqslant \mathbb{I}\left[y_i x_i^T \widehat{w}w_{(i)} \leqslant 0\right] \,, \tag{8.9}$$

where $\widehat{w}w_{(i)}$ is the classifier obtained by leaving out the i^{th} point from the training set. Intuitively, this condition states that the train error at a training point is less than the leave-one-out error at that point, i.e. the error obtained by removing that point and re-training. Using Condition 8.4.1, we then relate the average leave-one-out error (over the index i of the RHS in (8.9)) to the population error on the mislabeled distribution to obtain an inequality similar to (8.3). □

Extensions to kernel regression Since the result in Theorem 8.4.2 does not impose any regularity conditions on the underlying distribution over $\mathcal{X} \times \mathcal{Y}$, our guarantees extend straightforwardly to kernel regression by using the transformation $x \to \phi(x)$ for some feature transform function ϕ. Furthermore, recent literature has pointed out a concrete connection between neural networks and kernel regression with the so-called *Neural Tangent Kernel* (NTK) which holds in a certain regime where weights do not change much during training (Chizat et al., 2019; Du et al., 2019; 2018; Jacot et al., 2018). Using this concrete correspondence, our bounds on the clean population error (Theorem 8.4.2) extend to wide neural networks operating in the NTK regime.

Extensions to early stopped GD Often in practice, gradient descent is stopped early. We now provide theoretical evidence that our guarantees may continue to hold for an early stopped GD iterate. Concretely, we show that in expectation, the outputs of the GD iterates are close to that of a problem with data-independent regularization (as considered in Theorem 8.3.6). First, we introduce some notation. By $\mathcal{L}_S(w)$, we denote the objective in (8.6) with $\lambda = 0$. Consider the GD iterates defined in (8.7). Let $\widetilde{w}_\lambda = \arg\min_w \mathcal{L}_\mathcal{S}(w; \lambda)$. Define $f_t(x) := f(x; w_t)$ as the solution at the t^{th} iterate and $\widetilde{f}_\lambda(x) := f(x; \widetilde{w}_\lambda)$ as the regularized solution. Let κ be the condition number of the population covariance matrix and let s_{min} be the minimum positive singular value of the empirical covariance matrix.

Proposition 8.4.3 (informal)**.** *For $\lambda = \frac{1}{t\eta}$, we have*

$$\mathbb{E}_{x\sim\mathcal{D}_\mathcal{X}}\left[\left(f_t(x) - \widetilde{f}_\lambda(x)\right)^2\right] \leqslant c(t,\eta) \cdot \mathbb{E}_{x\sim\mathcal{D}_\mathcal{X}}\left[f_t(x)^2\right] \,,$$

where $c(t,\eta) \approx \kappa \cdot \min(0.25, \frac{1}{s_{min}^2 t^2 \eta^2})$. An equivalent guarantee holds for a point x sampled from the training data.

The proposition above states that for large enough t, GD iterates stay close to a regularized solution with data-independent regularization constant. Together with our guarantees in Theorem 8.4.2 for regularization solution with $\lambda = \frac{1}{t\eta}$, Proposition 8.4.3 shows that our guarantees with RATT may hold on early stopped GD. See the formal result in App. G.2.4.

Remark Proposition 8.4.3 only bounds the expected squared difference between the t^{th} gradient descent iterate and a corresponding regularized solution. The expected squared difference and the expected difference of classification errors (what we wish to bound) are not related, in general. However, they can be related under standard low-noise (margin) assumptions. For instance, under the Tsybakov noise condition (Tsybakov et al., 1997; Yao et al., 2007), we can lower-bound the expression on the LHS of Proposition 8.4.3 with the difference of expected classification error.

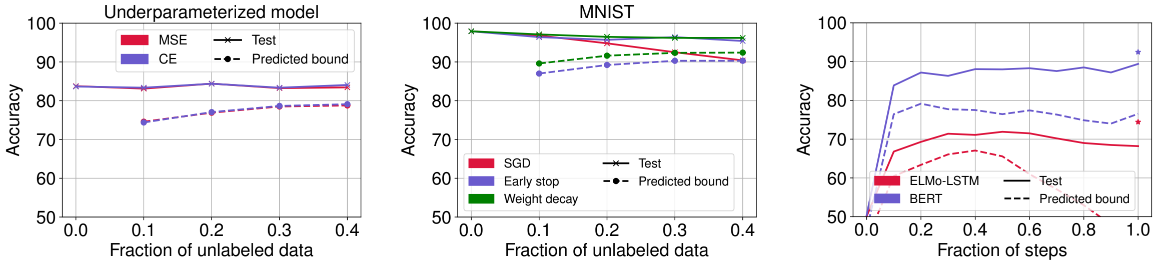
Extensions to deep learning Note that the main lemma underlying our bound on (clean) population error states that when training on a mixture of clean and randomly labeled data, we obtain a classifier whose empirical error on the mislabeled training data is lower than its population error on the distribution of mislabeled data. We prove this for ERM on 0-1 loss (Lemma 8.3.2). For linear models (and networks in NTK regime), we obtained this result by assuming hypothesis stability and relating training error at a datum with the leave-one-out error (Theorem 8.4.2). However, to extend our bound to deep models we must assume that training on the mixture of random and clean data leads to overfitting on the random mixture. Formally:

Assumption 2. *Let $\widehat{w}f$ be a model obtained by training with an algorithm $\mathcal{A}$ on a mixture of clean data S and randomly labeled data $\widetilde{S}$. Then with probability $1-\delta$ over the random draws of mislabeled data $\widetilde{S}_M$, we assume that the following condition holds:*

$$\mathcal{E}_{\widetilde{S}_M}(\widehat{w}f) \leqslant \mathcal{E}_{\mathcal{D}'}(\widehat{w}f) + c\sqrt{\frac{\log(1/\delta)}{2m}}\,,$$

for a fixed constant $c > 0$.

Under Assumption 2, our results in Theorem 8.3.1, 8.3.6 and 8.3.7 extend beyond ERM with the 0-1 loss to general learning algorithms. We include the formal result in App. G.2.5. Note that given the ability of neural networks to interpolate the data, this assumption seems uncontroversial in the later stages of training. Moreover, concerning the early phases of training, recent research has shown that learning dynamics for complex deep networks resemble those for linear models (Hu et al., 2020; Nakkiran et al., 2019), much like the wide neural networks that we do analyze. Together, these arguments help to justify Assumption 2 and hence, the applicability of our bound in deep learning. Motivated by our analysis on linear models trained with gradient descent, we discuss conditions in App. G.2.6 which imply Assumption 2 for constant values $\delta > 0$. In the next section, we empirically demonstrate applicability of our bounds for deep models.



(a) (b) (c)

Figure 8.2: We plot the accuracy and corresponding bound (RHS in (8.1)) at $\delta = 0.1$. for binary classification tasks. Results aggregated over 3 seeds. (a) Accuracy vs fraction of unlabeled data (w.r.t clean data) in the toy setup with a linear model trained with GD. (b) Accuracy vs fraction of unlabeled data for a 2-layer wide network trained with SGD on binary MNIST. With SGD and no regularization (red curve in (b)), we interpolate the training data and hence the predicted lower bound is 0. However, with early stopping (or weight decay) we obtain tight guarantees. (c) Accuracy vs gradient iteration on IMDb dataset with unlabeled fraction fixed at 0.2. In plot (c), '*' denotes the best test accuracy with the same hyperparameters and training only on clean data. See App. G.3 for exact hyperparameter values.

8.5 Empirical Study and Implications

Having established our framework theoretically, we now demonstrate its utility experimentally. First, for linear models and wide networks in the NTK regime where our guarantee holds, we confirm that our bound is not only valid, but closely tracks the generalization error. Next, we show that in practical deep learning settings, optimizing cross-entropy loss by SGD, the expression for our (0-1) ERM bound nevertheless tracks test performance closely and in numerous experiments on diverse models and datasets is never violated empirically.

Datasets To verify our results on linear models, we consider a toy dataset, where the class conditional distribution $p(x|y)$ for each label is Gaussian. For binary tasks, we use binarized CIFAR-10 (first 5 classes vs rest) (Krizhevsky and Hinton, 2009), binary MNIST (0-4 vs 5-9) (LeCun et al., 1998) and IMDb sentiment analysis dataset (Maas et al., 2011). For multiclass setup, we use MNIST and CIFAR-10.

Architectures To simulate the NTK regime, we experiment with 2-layered wide networks both (i) with the second layer fixed at random initialization; (ii) and updating both layers' weights. For vision datasets (e.g., MNIST and CIFAR10), we consider (fully connected) multilayer perceptrons (MLPs) with ReLU activations and ResNet18 (He et al., 2016). For the IMDb dataset, we train Long Short-Term Memory Networks (LSTMs; Hochreiter and Schmidhuber (1997)) with ELMo embeddings (Peters et al., 2018) and fine-tune an off-the-shelf uncased BERT model (Devlin et al., 2019; Wolf et al., 2020).

Methodology To bound the population error, we require access to both clean and unlabeled data. For toy datasets, we obtain unlabeled data by sampling from the underlying distribution over $\mathcal{X}$. For image and text datasets, we hold out a small fraction of the clean training data and discard their labels to simulate unlabeled data. We use the random labeling procedure described in Sec. 12.2. After augmenting clean training data with randomly labeled data, we train in the standard fashion. See App. G.3 for experimental details.

Underparameterized linear models On toy Gaussian data, we train linear models with GD to minimize cross-entropy loss and mean squared error. Varying the fraction of randomly labeled data we observe that the accuracy on clean unseen data is barely impacted (Fig. 8.2(a)). This highlights that in low dimensional models adding randomly labeled data with the clean dataset (in toy setup) has minimal effect on the performance on unseen clean data. Moreover, we find that RATT offers a tight lower bound on the unseen clean data accuracy. We observe the same behavior with Stochastic Gradient Descent (SGD) training (ref. App. G.3). Observe that the predicted bound goes up as the fraction of unlabeled data increases. While the accuracy as dictated by the dominating term in the RHS of (8.2) decreases with an increase in the fraction of unlabeled data, we observe a relatively sharper decrease in $\mathcal{O}_p\left(1/\sqrt{m}\right)$ term of the bound, leading to an overall increase in the predicted accuracy bound. In this toy setup, we also evaluated a kernel regression bound from Bartlett and Mendelson (2002) (Theorem 21), however, the predicted kernel regression bound remains vacuous.

Wide Nets Next, we consider MNIST binary classification with a wide 2-layer fully-connected network. In experiments with SGD training on MSE loss without early stopping or weight decay regularization, we find that adding extra randomly label data hurts the unseen clean performance (Fig. 8.2(b)). Additionally, due to the perfect fit on the training data, our bound is rendered vacuous. However, with early stopping (or weight decay), we observe close to zero performance difference with additional randomly labeled data. Alongside, we obtain tight bounds on the accuracy on unseen clean data paying only a small price to negligible for incorporating randomly labeled data. Similar results hold for SGD and GD and when cross-entropy loss is substituted for MSE (ref. App. G.3).

Deep Nets We verify our findings on (i) ResNet-18 and 5-layer MLPs trained with binary CIFAR (Fig. 8.1); and (ii) ELMo-LSTM and BERT-Base models fine-tuned on the IMDb dataset (Fig. 8.2(c)). See App. G.3 for additional results with deep models on binary MNIST. We fix the amount of unlabeled data at 20% of the clean dataset size and train all models with standard hyperparameters. Consistently, we find that our predicted bounds are never violated in practice. And as training proceeds, the fit on the mislabeled data increases with perfect overfitting in the interpolation regime rendering our bounds vacuous. However, with early stopping, our bound predicts test performance closely. For example, on IMDb dataset with BERT fine-tuning we predict 79.8 as the accuracy of the classifier, when the true performance is 88.04 (and the best achievable performance on unseen data is 92.45).

Dataset	Model	Pred. Acc	Test Acc.	Best Acc.
MNIST	MLP	93.1	97.4	97.9
	ResNet	96.8	98.8	98.9
CIFAR10	MLP	48.4	54.2	60.0
	ResNet	76.4	88.9	92.3

Table 8.1: Results on multiclass classification tasks. With pred. acc. we refer to the dominating term in RHS of (8.5). At the given sample size and $\delta = 0.1$, the remaining term evaluates to 30.7, decreasing our predicted accuracy by the same. We note that test acc. denotes the corresponding accuracy on unseen clean data. Best acc. is the best achievable accuracy with just training on just the clean data (and same hyperparamters except the stopping point). Note that across all tasks our predicted bound is tight and the gap between the best accuracy and test accuracy is small. Exact hyperparameters are included in App. G.3.

Additionally, we observe that our method tracks the performance from the beginning of the training and not just towards the end.

Finally, we verify our multiclass bound on MNIST and CIFAR10 with deep MLPs and ResNets (see results in Table 8.1 and per-epoch curves in App. G.3). As before, we fix the amount of unlabeled data at 20% of the clean dataset to minimize cross-entropy loss via SGD. In all four settings, our bound predicts non-vacuous performance on unseen data. In App. G.3, we investigate our approach on CIFAR100 showing that even though our bound grows pessimistic with greater numbers of classes, the error on the mislabeled data nevertheless tracks population accuracy.

8.6 Discussion and Connections to Prior Work

Implicit bias in deep learning Several recent lines of research attempt to explain the generalization of neural networks despite massive overparameterization via the *implicit bias* of gradient descent (Chizat and Bach, 2020; Gunasekar et al., 2018a;b; Ji and Telgarsky, 2019; Soudry et al., 2018). Noting that even for overparameterized linear models, there exist multiple parameters capable of overfitting the training data (with arbitrarily low loss), of which some generalize well and others do not, they seek to characterize the favored solution. Notably, Soudry et al. (2018) find that for linear networks, gradient descent converges (slowly) to the max margin solution. A complementary line of work focuses on the early phases of training, finding both empirically (Arpit et al., 2017; Rolnick et al., 2017) and theoretically (Arora et al., 2019a; Li et al., 2020; Liu et al., 2020) that even in the presence of a small amount of mislabeled data, gradient descent is biased to fit the clean data first during initial phases of training. However, to best our knowledge, no prior work leverages this phenomenon to obtain generalization guarantees on the clean data, which is the primary focus of our work. Our method exploits this phenomenon to produce non-vacuous generalization bounds. Even when we cannot prove *a priori* that models will fit the clean data well while performing badly on the mislabeled data, we can observe that it indeed happens (often in practice), and thus, *a posteriori*, provide tight bounds on the population error. Moreover, by using regularizers like early stopping or weight decay, we can accentuate this phenomenon, enabling our framework to provide even tighter guarantees.

Generalization bounds Conventionally, generalization in machine learning has been studied through the lens of uniform convergence bounds (Blumer et al., 1989; Vapnik, 1999). Representative works on understanding generalization in overparameterized networks within this framework include Allen-Zhu et al. (2019a); Arora et al. (2018); Bartlett et al. (2017); Dziugaite and Roy (2017); Li and Liang (2018); Nagarajan and Kolter (2019a); Neyshabur et al. (2015; 2017a;b; 2018); Zhou et al. (2018). However, uniform convergence based bounds typically remain numerically loose relative to the true generalization error. Several works have also questioned the ability of uniform convergence based approaches to explain generalization in overparameterized models (Nagarajan and Kolter, 2019b; Zhang et al., 2017). Subsequently, recent works have proposed unconventional ways to derive generalization bounds (Negrea et al., 2020; Zhou et al., 2020). In a similar spirit, we take departure from complexity-based approaches to generalization bounds in our work. In particular, we leverage unlabeled data to derive a post-hoc generalization bound. Our work provides guarantees on overparameterized networks by using early stopping or weight decay regularization, preventing a perfect fit on the training data. Notably, in our framework, the model can perfectly fit the clean portion of the data, so long as they nevertheless fit the mislabeled data poorly.

Leveraging noisy data to provide generalization guarantees In parallel work, Bansal et al. (2020) presented an upper bound on the generalization gap of linear classifiers trained on representations learned via self-supervision. Under certain noise-robustness and rationality assumptions on the training procedure, the authors obtained bounds dependent on the complexity of the linear classifier and independent of the complexity of representations. By contrast, we present generalization bounds for supervised learning that are non-vacuous by virtue of the early learning phenomenon. While both frameworks highlight how robustness to random label corruptions can be leveraged to obtain bounds that do not depend directly on the complexity of the underlying hypothesis class, our framework, methodology, claims, and generalization results are very different from theirs.

Other related work. A long line of work relates early stopped GD to a corresponding regularized solution (Ali et al., 2018; 2020; Friedman and Popescu, 2003; Neu and Rosasco, 2018; Suggala et al., 2018; Yao et al., 2007). In the most relevant work, Ali et al. (2018) and Suggala et al. (2018) address a regression task, theoretically relating the solutions of early-stopped GD and a regularized problem, obtained with a data-independent regularization coefficient. Towards understanding generalization numerous stability conditions have been discussed (Bousquet and Elisseeff, 2002; Kearns and Ron, 1999; Mukherjee et al., 2006; Shalev-Shwartz et al., 2010). Hardt et al. (2016) studies the uniform stability property to obtain generalization guarantees with early-stopped SGD. While we assume a benign stability condition to relate leave-one-out performance with population error, we do not rely on any stability condition that implies generalization.

8.7 Conclusion and Future work

Our work introduces a new approach for obtaining generalization bounds that do not directly depend on the underlying complexity of the model class. For linear models, we provably obtain a bound in terms of the fit on randomly labeled data added during training. Our findings raise a number of questions to be explored next. While our empirical findings and theoretical results with 0-1 loss hold absent further assumptions and shed light on why the bound may apply for more general models, we hope to extend our proof that overfitting (in terms classification error) to the finite sample of mislabeled data occurs with SGD training on broader classes of models and loss functions. We hope to build on some early results (Hu et al., 2020; Nakkiran et al., 2019) which provide evidence that deep models behave like linear models in the early phases of training. We also wish to extend our framework to the interpolation regime. Since many important aspects of neural network learning take place within early epochs (Achille et al., 2017; Frankle et al., 2020), including gradient dynamics converging to very small subspace (Gur-Ari et al., 2018), we might imagine operationalizing our bounds in the interpolation regime by discarding the randomly labeled data after initial stages of training.

Chapter 9

Leveraging Unlabeled Data to Predict Out-of-Distribution Performance

Based on Garg et al. (2022b): Saurabh Garg, Sivaraman Balakrishnan, Zachary C. Lipton, Behnam Neyshabur, and Hanie Sedghi. Leveraging unlabeled data to predict out-of-distribution performance. Internation Conference on Learning Representations (ICLR), 2022.

Abstract

Real-world machine learning deployments are characterized by mismatches between the source (training) and target (test) distributions that may cause performance drops. In this chapter, we investigate methods for predicting the target domain accuracy using only labeled source data and unlabeled target data. We propose Average Thresholded Confidence (ATC), a practical method that learns a *threshold* on the model's confidence, predicting accuracy as the fraction of unlabeled examples for which model confidence exceeds that threshold. ATC outperforms previous methods across several model architectures, types of distribution shifts (e.g., due to synthetic corruptions, dataset reproduction, or novel subpopulations), and datasets (WILDS, ImageNet, BREEDS, CIFAR, and MNIST). In our experiments, ATC estimates target performance 2–4× more accurately than prior methods. We also explore the theoretical foundations of the problem, proving that, in general, identifying the accuracy is just as hard as identifying the optimal predictor and thus, the efficacy of any method rests upon (perhaps unstated) assumptions on the nature of the shift. Finally, analyzing our method on some toy distributions, we provide insights concerning when it works. Code is available at this url.

9.1 Introduction

The previous chapter focused on assessing model performance under distribution shift. In this and the next chapter, we will focus on predicting performance under distribution shift. As demonstrated in the previous chapter, we can obtain tight generalization bounds on in-distribution performance with mild assumptions that deep models overfit. We begin by investigating two questions: (i) the precise conditions under which we can estimate a classifier's target-domain accuracy; and (ii) which methods are most practically useful. To begin, the straightforward way to assess the performance of a model under distribution shift would be to collect labeled (target domain) examples and then to evaluate the model on that data. However, collecting fresh labeled data from the target distribution is prohibitively expensive and time-consuming, especially if the target distribution is non-stationary. Hence, instead of using labeled data, we aim to use unlabeled data from the target distribution, that is comparatively abundant, to predict model performance. Note that in this work, our focus is *not* to improve performance on the target but, rather, to estimate the accuracy on the target for a given classifier.

Recently, numerous methods have been proposed for this purpose (Chen et al., 2021b; Deng and Zheng, 2021; Deng et al., 2021; Guillory et al., 2021; Jiang et al., 2021). These methods either require calibration on the target domain to yield consistent estimates (Guillory et al., 2021; Jiang et al., 2021) or additional labeled data from several target domains to learn a linear regression function on a distributional distance that then predicts model performance (Deng and Zheng, 2021; Deng et al., 2021; Guillory et al., 2021). However, methods that require calibration on the target domain typically yield poor estimates since deep models trained and calibrated on source data are not, in general, calibrated on a (previously unseen) target domain (Ovadia et al., 2019). Besides, methods that leverage labeled data from target domains rely on the fact that unseen target domains exhibit strong linear correlation with seen target domains on the underlying distance measure and, hence, can be rendered ineffective when such target domains with labeled data are unavailable (in Sec. 9.5.1 we demonstrate such a failure on a real-world distribution shift problem). Therefore, throughout the chapter, we assume access to labeled source data and only unlabeled data from target domain(s).

In this work, we first show that absent assumptions on the source classifier or the nature of the shift, no method of estimating accuracy will work generally (even in non-contrived settings). To estimate accuracy on target domain *perfectly*, we highlight that even given perfect knowledge of the labeled source distribution (i.e., $p_s(x, y)$) and unlabeled target distribution (i.e., $p_t(x)$), we need restrictions on the nature of the shift such that we can uniquely identify the target conditional $p_t(y|x)$. Thus, in general, identifying the accuracy of the classifier is as hard as identifying the optimal predictor.

Second, motivated by the superiority of methods that use maximum softmax probability (or logit) of a model for Out-Of-Distribution (OOD) detection (Hendrycks and Gimpel, 2017; Hendrycks et al., 2019), we propose a simple method that leverages softmax probability to predict model performance. Our method, Average Thresholded Confidence (ATC),

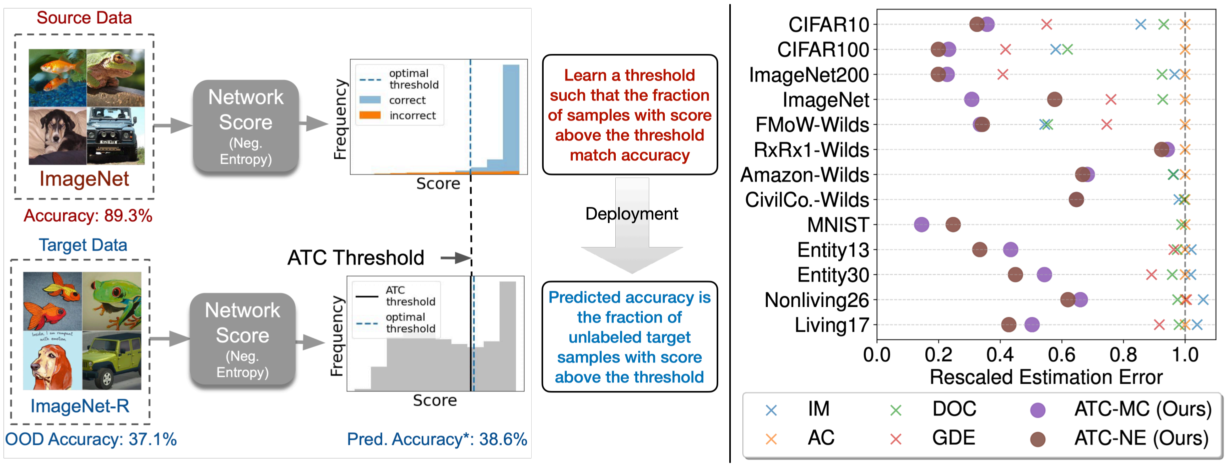


Figure 9.1: *Illustration of our proposed method ATC.* **Left**: using source domain validation data, we identify a *threshold* on a score (e.g. negative entropy) computed on model confidence such that fraction of examples above the threshold matches the validation set accuracy. ATC estimates accuracy on unlabeled target data as the fraction of examples with the score above the threshold. Interestingly, this threshold yields accurate estimates on a wide set of target distributions resulting from natural and synthetic shifts. **Right**: Efficacy of ATC over previously proposed approaches on our testbed with a post-hoc calibrated model. To obtain errors on the same scale, we rescale all errors with Average Confidence (AC) error. Lower estimation error is better. See Table 9.1 for exact numbers and comparison on various types of distribution shift. See Sec. 9.5 for details on our testbed.

learns a threshold on a score (e.g., maximum confidence or negative entropy) of model confidence on validation source data and predicts target domain accuracy as the fraction of unlabeled target points that receive a score above that threshold. ATC selects a threshold on validation source data such that the fraction of source examples that receive the score above the threshold match the accuracy of those examples. Our primary contribution in ATC is the proposal of obtaining the threshold and observing its efficacy on (practical) accuracy estimation. Importantly, our work takes a step forward in positively answering the question raised in Deng and Zheng (2021); Deng et al. (2021) about a practical strategy to select a threshold that enables accuracy prediction with thresholded model confidence.

ATC is simple to implement with existing frameworks, compatible with arbitrary model classes, and dominates other contemporary methods. Across several model architectures on a range of benchmark vision and language datasets, we verify that ATC outperforms prior methods by at least 2–4× in predicting target accuracy on a variety of distribution shifts. In particular, we consider shifts due to common corruptions (e.g., ImageNet-C), natural distribution shifts due to dataset reproduction (e.g., ImageNet-v2, ImageNet-R), shifts due to novel subpopulations (e.g., BREEDS), and distribution shifts faced in the wild (e.g., WILDS).

As a starting point for theory development, we investigate ATC on a simple toy model that models distribution shift with varying proportions of the population with spurious features, as in Nagarajan et al. (2020). Finally, we note that although ATC achieves superior performance in our empirical evaluation, like all methods, it must fail (returns inconsistent estimates) on certain types of distribution shifts, per our impossibility result.

9.2 Prior Work

Out-of-distribution detection. The main goal of OOD detection is to identify previously unseen examples, i.e., samples out of the support of training distribution. To accomplish this, modern methods utilize confidence or features learned by a deep network trained on some source data. Geifman and El-Yaniv (2017); Hendrycks and Gimpel (2017) used the confidence score of an (already) trained deep model to identify OOD points. Lakshminarayanan et al. (2016) use entropy of an ensemble model to evaluate prediction uncertainty on OOD points. To improve OOD detection with model confidence, Liang et al. (2018) propose to use temperature scaling and input perturbations. Jiang et al. (2018) propose to use scores based on the relative distance of the predicted class to the second class. Recently, residual flow-based methods were used to obtain a density model for OOD detection (Zhang et al., 2020). Ji et al. (2021) proposed a method based on subfunction error bounds to compute unreliability per sample. Refer to Ji et al. (2021); Ovadia et al. (2019) for an overview and comparison of methods for prediction uncertainty on OOD data.

Predicting model generalization. Relevant to our work are methods for predicting the error of a classifier on OOD data based on unlabeled data from the target (OOD) domain. These methods can be characterized into two broad categories: (i) Methods which explicitly predict correctness of the model on individual unlabeled points (Chen et al., 2021a; Deng and Zheng, 2021; Deng et al., 2021; Jiang et al., 2021); and (ii) Methods which directly obtain an estimate of error with unlabeled OOD data without making a point-wise prediction (Chen et al., 2021b; Chuang et al., 2020; Guillory et al., 2021).

To achieve a consistent estimate of the target accuracy, Guillory et al. (2021); Jiang et al. (2021) require calibration on target domain. However, these methods typically yield poor estimates as deep models trained and calibrated on some source data are seldom calibrated on previously unseen domains (Ovadia et al., 2019). Additionally, Deng and Zheng (2021); Guillory et al. (2021) derive model-based distribution statistics on unlabeled target set that correlate with the target accuracy and propose to use a subset of *labeled* target domains to learn a (linear) regression function that predicts model performance. However, there are two drawbacks with this approach: (i) the correlation of these distribution statistics can vary substantially as we consider different nature of shifts (refer to Sec. 9.5.1, where we empirically demonstrate this failure); (ii) even if there exists a (hypothetical) statistic with strong correlations, obtaining labeled target domains (even simulated ones) with strong correlations would require significant *a priori* knowledge about the nature of shift that, in general, might not be available before models are deployed in the wild. Nonetheless, in our work, we only assume access to labeled data from the source domain presuming no access to labeled target domains or information about how to simulate them.

Moreover, unlike the parallel work of Deng et al. (2021), we do not focus on methods that alter the training on source data to aid accuracy prediction on the target data. Chen et al. (2021b) propose an importance re-weighting based approach that leverages (additional) information about the axis along which distribution is shifting in form of "slicing functions". In our work, we make comparisons with importance re-weighting baseline from Chen et al. (2021b) as we do not have any additional information about the axis along which the distribution is shifting.

9.3 Problem Setup

Notation. By $\|\cdot\|$, and $\langle\cdot,\cdot\rangle$ we denote the Euclidean norm and inner product, respectively. For a vector $v \in \mathbb{R}^d$, we use v_j to denote its j^{th} entry, and for an event E we let $\mathbb{I}\left[E\right]$ denote the binary indicator of the event.

Suppose we have a multi-class classification problem with the input domain $\mathcal{X} \subseteq \mathbb{R}^d$ and label space $\mathcal{Y} = \{1, 2, \ldots, k\}$. For binary classification, we use $\mathcal{Y} = \{0, 1\}$. By $\mathcal{D}^{\text{S}}$ and $\mathcal{D}^{\text{T}}$, we denote source and target distribution over $\mathcal{X} \times \mathcal{Y}$. For distributions $\mathcal{D}^{\text{S}}$ and $\mathcal{D}^{\text{T}}$, we define p_{S} or p_{T} as the corresponding probability density (or mass) functions. A dataset $S := \{(x_i, y_i)\}_{i=1}^n \sim (\mathcal{D}^{\text{S}})^n$ contains n points sampled i.i.d. from $\mathcal{D}^{\text{S}}$. Let $\mathcal{F}$ be a class of hypotheses mapping $\mathcal{X}$ to Δ^{k-1} where Δ^{k-1} is a simplex in k dimensions. Given a classifier $f \in \mathcal{F}$ and datum (x, y), we denote the 0-1 error (i.e., classification error) on that point by $\mathcal{E}(f(x), y) := \mathbb{I}\left[y \notin \arg\max_{j \in \mathcal{Y}} f_j(x)\right]$. Given a model $f \in \mathcal{F}$, our goal in this work is to understand the performance of f on $\mathcal{D}^{\text{T}}$ without access to labeled data from $\mathcal{D}^{\text{T}}$. Note that our goal is not to adapt the model to the target data. Concretely, we aim to predict accuracy of f on $\mathcal{D}^{\text{T}}$. Throughout this paper, we assume we have access to the following: (i) model f; (ii) previously-unseen (validation) data from $\mathcal{D}^{\text{S}}$; and (iii) unlabeled data from target distribution $\mathcal{D}^{\text{T}}$.

9.3.1 Accuracy Estimation: Possibility and Impossibility Results

First, we investigate the question of when it is possible to estimate the target accuracy of an arbitrary classifier, even given knowledge of the full source distribution $p_s(x, y)$ and target marginal $p_t(x)$. Absent assumptions on the nature of shift, estimating target accuracy is impossible. Even given access to $p_s(x, y)$ and $p_t(x)$, the problem is fundamentally unidentifiable because $p_t(y|x)$ can shift arbitrarily. In the following proposition, we show that absent assumptions on the classifier f (i.e., when f can be any classifier in the space of all classifiers on $\mathcal{X}$), we can estimate accuracy on the target data iff assumptions on the nature of the shift, together with $p_s(x, y)$ and $p_t(x)$, uniquely identify the (unknown) target conditional $p_t(y|x)$. We relegate proofs from this section to App. H.1.

Proposition 9.3.1. *Absent further assumptions, accuracy on the target is identifiable iff $p_t(y|x)$ is uniquely identified given $p_s(x, y)$ and $p_t(x)$.*

Proposition 9.3.1 states that we need enough constraints on nature of shift such that $p_s(x, y)$ and $p_t(x)$ identifies unique $p_t(y|x)$. It also states that under some assumptions on the nature of the shift, we can hope to estimate the model's accuracy on target data. We will illustrate this on two common assumptions made in domain adaptation literature: (i) covariate shift (Heckman, 1977; Shimodaira, 2000) and (ii) label shift (Lipton et al., 2018b; Saerens et al., 2002; Zhang et al., 2013). Under covariate shift assumption, that the target marginal support $\mathbf{supp}(p_t(x))$ is a subset of the source marginal support $\mathbf{supp}(p_s(x))$ and that the conditional distribution of labels given inputs does not change within support, i.e., $p_s(y|x) = p_t(y|x)$, which, trivially, identifies a unique target conditional $p_t(y|x)$. Under label shift, the reverse holds, i.e., the class-conditional distribution does not change ($p_s(x|y) = p_t(x|y)$) and, again, information about $p_t(x)$ uniquely determines the target conditional $p_t(y|x)$ (Garg et al., 2020b; Lipton et al., 2018b). In these settings, one can estimate an arbitrary classifier's accuracy on the target domain either by using importance re-weighting with the ratio $p_t(x)/p_s(x)$ in case of covariate shift or by using importance re-weighting with the ratio $p_t(y)/p_s(y)$ in case of label shift. While importance ratios in the former case can be obtained directly when $p_t(x)$ and $p_s(x)$ are known, the importance ratios in the latter case can be obtained by using techniques from Alexandari et al. (2021); Azizzadenesheli et al. (2019); Lipton et al. (2018b).

As a corollary of Proposition 9.3.1, we now present a simple impossibility result, demonstrating that no single method can work for all families of distribution shift.

Corollary 9.3.2. *Absent assumptions on the classifier f, no method of estimating accuracy will work in all scenarios, i.e., for different nature of distribution shifts.*

Intuitively, this result states that every method of estimating accuracy on target data is tied up with some assumption on the nature of the shift and might not be useful for estimating accuracy under a different assumption on the nature of the shift. For illustration, consider a setting where we have access to distribution $p_s(x, y)$ and $p_t(x)$. Additionally, assume that the distribution can shift only due to covariate shift or label shift without any knowledge about which one. Then Corollary 9.3.2 says that it is impossible to have a single method that will simultaneously for both label shift and covariate shift as in the following example (we spell out the details in App. H.1):

Example 1. Assume binary classification with $p_s(x) = \alpha \cdot \phi(\mu_1) + (1 - \alpha) \cdot \phi(\mu_2)$, $p_s(x|y = 0) = \phi(\mu_1)$, $p_s(x|y = 1) = \phi(\mu_2)$, and $p_t(x) = \beta \cdot \phi(\mu_1) + (1 - \beta) \cdot \phi(\mu_2)$ where $\phi(\mu) = \mathcal{N}(\mu, 1)$, $\alpha, \beta \in (0, 1)$, and $\alpha \neq \beta$. Error of a classifier f on target data is given by $\mathcal{E}_1 = \mathbb{E}_{(x,y)\sim p_s(x,y)}\left[\frac{p_t(x)}{p_s(x)}\mathbb{I}\left[f(x) \neq y\right]\right]$ under covariate shift and by $\mathcal{E}_2 = \mathbb{E}_{(x,y)\sim p_s(x,y)}\left[\left(\frac{\beta}{\alpha}\mathbb{I}\left[y = 0\right] + \frac{1-\beta}{1-\alpha}\mathbb{I}\left[y = 1\right]\right)\mathbb{I}\left[f(x) \neq y\right]\right]$ under label shift. In App. H.1, we show that $\mathcal{E}_1 \neq \mathcal{E}_2$ for all f. Thus, given access to $p_s(x, y)$, and $p_t(x)$, any method that consistently estimates error of a classifer under covariate shift will give an incorrect estimate of error under label shift and vice-versa. The reason is that the same $p_t(x)$ and $p_s(x, y)$ can correspond to error $\mathcal{E}_1$ (under covariate shift) or error $\mathcal{E}_2$ (under label shift) and determining which scenario one faces requires further assumptions on the nature of shift.

9.4 Predicting accuracy with Average Thresholded Confidence

In this section, we present our method ATC that leverages a black box classifier f and (labeled) validation source data to predict accuracy on target domain given access to unlabeled target data. Throughout the discussion, we assume that the classifier f is fixed.

Before presenting our method, we introduce some terminology. Define a score function $s : \Delta^{k-1} \to \mathbb{R}$ that takes in the softmax prediction of the function f and outputs a scalar. We want a score function such that if the score function takes a high value at a datum (x, y) then f is likely to be correct. In this work, we explore two such score functions: (i) Maximum confidence, i.e., $s(f(x)) = \max_{j\in\mathcal{Y}} f_j(x)$; and (ii) Negative Entropy, i.e., $s(f(x)) = \sum_j f_j(x) \log(f_j(x))$. Our method identifies a threshold t on source data $\mathcal{D}^{\text{S}}$ such that the expected number of points that obtain a score less than t match the error of f on $\mathcal{D}^{\text{S}}$, i.e.,

$$\mathbb{E}_{x\sim\mathcal{D}^{\text{S}}}\left[\mathbb{I}\left[s(f(x)) < t\right]\right] = \mathbb{E}_{(x,y)\sim\mathcal{D}^{\text{S}}}\left[\mathbb{I}\left[\arg\max_{j\in\mathcal{Y}} f_j(x) \neq y\right]\right], \tag{9.1}$$

and then our error estimate $\text{ATC}_{\mathcal{D}^{\text{T}}}(s)$ on the target domain $\mathcal{D}^{\text{T}}$ is given by the expected number of target points that obtain a score less than t, i.e.,

$$\text{ATC}_{\mathcal{D}^{\text{T}}}(s) = \mathbb{E}_{x\sim\mathcal{D}^{\text{T}}}\left[\mathbb{I}\left[s(f(x)) < t\right]\right]. \tag{9.2}$$

In short, in (9.1), ATC selects a threshold on the score function such that the error in the source domain matches the expected number of points that receive a score below t and in (9.2), ATC predicts error on the target domain as the fraction of unlabeled points that obtain a score below that threshold t. Note that, in principle, there exists a different threshold t' on the target distribution $\mathcal{D}^{\text{T}}$ such that (9.1) is satisfied on $\mathcal{D}^{\text{T}}$. However, in our experiments, the same threshold performs remarkably well. The main empirical contribution of our work is to show that the threshold obtained with (9.1) might be used effectively in conduction with modern deep networks in a wide range of settings to estimate error on the target data. In practice, to obtain the threshold with ATC, we minimize the difference between the expression on two sides of (9.1) using finite samples. In the next section, we show that ATC precisely predicts accuracy on the OOD data on the desired line $y = x$. In App. H.3, we discuss an alternate interpretation of the method and make connections with OOD detection methods.

9.5 Experiments

We now empirical evaluate ATC and compare it with existing methods. In each of our main experiment, keeping the underlying model fixed, we vary target datasets and make a prediction of the target accuracy with various methods given access to only unlabeled data

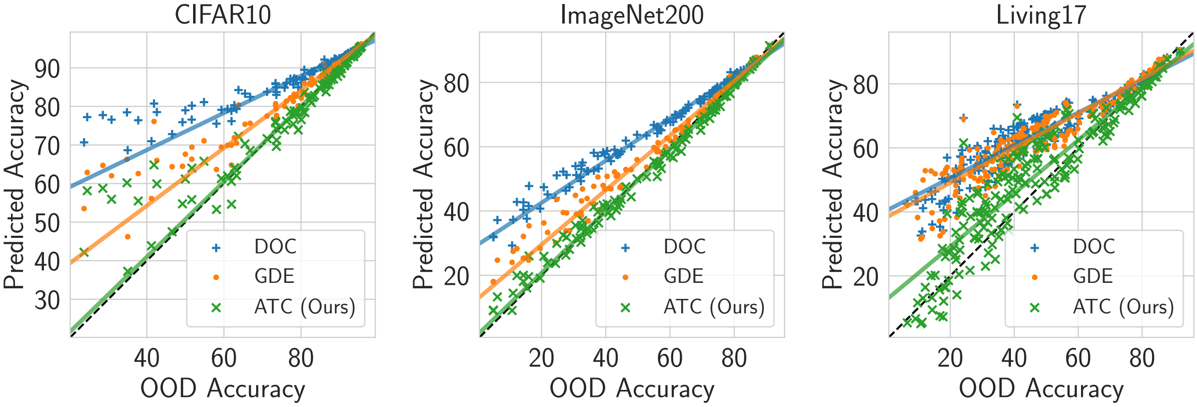


Figure 9.2: *Scatter plot of predicted accuracy versus (true) OOD accuracy.* Each point denotes a different OOD dataset, all evaluated with the same DenseNet121 model. We only plot the best three methods. With ATC (ours), we refer to ATC-NE. We observe that ATC significantly outperforms other methods and with ATC, we recover the desired line $y = x$ with a robust linear fit. Aggregated estimation error in Table 9.1 and plots for other datasets and architectures in App. H.8.

from the target. Unless noted otherwise, all models are trained only on samples from the source distribution with the main exception of pre-training on a different distribution. We use labeled examples from the target distribution to only obtain true error estimates.

Datasets. First, we consider synthetic shifts induced due to different visual corruptions (e.g., shot noise, motion blur etc.) under ImageNet-C (Hendrycks and Dietterich, 2019). Next, we consider natural shifts due to differences in the data collection process of ImageNet (Russakovsky et al., 2015), e.g, ImageNetv2 (Recht et al., 2019b). We also consider images with artistic renditions of object classes, i.e., ImageNet-R (Hendrycks et al., 2021b) and ImageNet-Sketch (Wang et al., 2019b). Note that renditions dataset only contains a subset 200 classes from ImageNet. To include renditions dataset in our testbed, we include results on ImageNet restricted to these 200 classes (which we call ImageNet-200) along with full ImageNet.

Second, we consider Breeds (Santurkar et al., 2021) to assess robustness to subpopulation shifts, in particular, to understand how accuracy estimation methods behave when novel subpopulations not observed during training are introduced. Breeds leverages class hierarchy in ImageNet to create 4 datasets Entity-13, Entity-30, Living-17, Non-living-26. We focus on natural and synthetic shifts as in ImageNet on same and different subpopulations in BREEDs. Third, from Wilds (Koh et al., 2021) benchmark, we consider FMoW-Wilds (Christie et al., 2018), RxRx1-Wilds (Taylor et al., 2019), Amazon-Wilds (Ni et al., 2019), CivilComments-Wilds (Borkan et al., 2019) to consider distribution shifts faced in the wild.

Finally, similar to ImageNet, we consider (i) synthetic shifts (CIFAR-10-C) due to common corruptions; and (ii) natural shift (i.e., CIFARv2 (Recht et al., 2018)) on CIFAR-10 (Krizhevsky and Hinton, 2009). On CIFAR-100, we just have synthetic shifts due to common corruptions. For completeness, we also consider natural shifts on MNIST (LeCun et al., 1998) as in the prior work (Deng and Zheng, 2021). We use three real shifted datasets, i.e., USPS (Hull, 1994), SVHN (Netzer et al., 2011a) and QMNIST (Yadav and Bottou, 2019). We give a detailed overview of our setup in App. H.6.

Architectures and Evaluation. For ImageNet, BREEDS, CIFAR, FMoW-WILDS, RxRx1-WILDS datasets, we use DenseNet121 (Huang et al., 2017) and ResNet50 (He et al., 2016) architectures. For Amazon-WILDS and CivilComments-WILDS, we fine-tune a DistilBERT-base-uncased (Sanh et al., 2019a) model. For MNIST, we train a fully connected multilayer perceptron. We use standard training with benchmarked hyperparameters. To compare methods, we report average absolute difference between the true accuracy on the target data and the estimated accuracy on the same unlabeled examples. We refer to this metric as Mean Absolute estimation Error (MAE). Along with MAE, we also show scatter plots to visualize performance at individual target sets. Refer to App. H.7 for additional details on the setup.

Methods With ATC-NE, we denote ATC with negative entropy score function and with ATC-MC, we denote ATC with maximum confidence score function. For all methods, we implement *post-hoc* calibration on validation source data with Temperature Scaling (TS; Guo et al. (2017)). Below we briefly discuss baselines methods compared in our work and relegate details to App. H.5.

Average Confidence (AC). Error is estimated as the expected value of the maximum softmax confidence on the target data, i.e, $\text{AC}_{\mathcal{D}^\text{T}} = \mathbb{E}_{x\sim\mathcal{D}^\text{T}}\left[\max_{j\in\mathcal{Y}} f_j(x)\right]$.

Difference Of Confidence (DOC). We estimate error on target by subtracting difference of confidences on source and target (as a surrogate to distributional distance Guillory et al. (2021)) from the error on source distribution, i.e, $\text{DOC}_{\mathcal{D}^\text{T}} = \mathbb{E}_{x\sim\mathcal{D}^\text{S}}\left[\mathbb{I}\left[\arg\max_{j\in\mathcal{Y}} f_j(x) \neq y\right]\right] + \mathbb{E}_{x\sim\mathcal{D}^\text{T}}\left[\max_{j\in\mathcal{Y}} f_j(x)\right] - \mathbb{E}_{x\sim\mathcal{D}^\text{S}}\left[\max_{j\in\mathcal{Y}} f_j(x)\right]$. This is referred to as DOC-Feat in (Guillory et al., 2021).

Importance re-weighting (IM). We estimate the error of the classifier with importance re-weighting of 0-1 error in the pushforward space of the classifier. This corresponds to MANDOLIN using one slice based on the underlying classifier confidence Chen et al. (2021b).

Generalized Disagreement Equality (GDE). Error is estimated as the expected disagreement of two models (trained on the same training set but with different randomization) on target data (Jiang et al., 2021), i.e., $\text{GDE}_{\mathcal{D}^\text{T}} = \mathbb{E}_{x\sim\mathcal{D}^\text{T}}\left[\mathbb{I}\left[f(x) \neq f'(x)\right]\right]$ where f and f' are the two models. Note that GDE requires two models trained independently, doubling the computational overhead while training.

9.5.1 Results

In Table 9.1, we report MAE results aggregated by the nature of the shift in our testbed. In Fig. 9.2 and Fig. 9.1(right), we show scatter plots for predicted accuracy versus OOD accuracy on several datasets. We include scatter plots for all datasets and parallel results

Dataset	Shift	IM		AC		DOC		GDE	ATC-MC (Ours)		ATC-NE (Ours)	
		Pre T	Post T	Pre T	Post T	Pre T	Post T	Post T	Pre T	Post T	Pre T	Post T
CIFAR10	Natural	6.60	5.74	9.88	6.89	7.25	6.07	4.77	3.21	3.02	2.99	**2.85**
	Synthetic	12.33	10.20	16.50	11.91	13.87	11.08	6.55	4.65	4.25	4.21	**3.87**
CIFAR100	Synthetic	13.69	11.51	23.61	13.10	14.60	10.14	9.85	5.50	**4.75**	**4.72**	4.94
ImageNet200	Natural	12.37	8.19	22.07	8.61	15.17	7.81	5.13	4.37	2.04	3.79	**1.45**
	Synthetic	19.86	12.94	32.44	13.35	25.02	12.38	5.41	5.93	3.09	5.00	**2.68**
ImageNet	Natural	7.77	6.50	18.13	6.02	8.13	5.76	6.23	3.88	2.17	2.06	**0.80**
	Synthetic	13.39	10.12	24.62	8.51	13.55	7.90	6.32	3.34	**2.53**	**2.61**	4.89
FMoW-WILDS	Natural	5.53	4.31	33.53	12.84	5.94	4.45	5.74	3.06	**2.70**	3.02	**2.72**
RxRx1-WILDS	Natural	5.80	5.72	7.90	4.84	5.98	5.98	6.03	4.66	**4.56**	**4.41**	**4.47**
Amazon-WILDS	Natural	2.40	2.29	8.01	2.38	2.40	2.28	17.87	1.65	**1.62**	**1.60**	**1.59**
CivilCom.-WILDS	Natural	12.64	10.80	16.76	11.03	13.31	10.99	16.65	**7.14**			
MNIST	Natural	18.48	15.99	21.17	14.81	20.19	14.56	24.42	5.02	**2.40**	3.14	3.50
Entity-13	Same	16.23	11.14	24.97	10.88	19.08	10.47	10.71	5.39	**3.88**	4.58	4.19
	Novel	28.53	22.02	38.33	21.64	32.43	21.22	20.61	13.58	10.28	12.25	**6.63**
Entity-30	Same	18.59	14.46	28.82	14.30	21.63	13.46	12.92	9.12	**7.75**	8.15	**7.64**
	Novel	32.34	26.85	44.02	26.27	36.82	25.42	23.16	17.75	14.30	15.60	**10.57**
Nonliving-26	Same	18.66	17.17	26.39	16.14	19.86	15.58	16.63	10.87	**10.24**	10.07	**10.26**
	Novel	33.43	31.53	41.66	29.87	35.13	29.31	29.56	21.70	20.12	19.08	**18.26**
Living-17	Same	12.63	11.05	18.32	10.46	14.43	10.14	9.87	4.57	**3.95**	**3.81**	4.21
	Novel	29.03	26.96	35.67	26.11	31.73	25.73	23.53	16.15	14.49	12.97	**11.39**

Table 9.1: *Mean Absolute estimation Error (MAE) results for different datasets in our setup grouped by the nature of shift.* 'Same' refers to same subpopulation shifts and 'Novel' refers novel subpopulation shifts. We include details about the target sets considered in each shift in Table H.1. Post T denotes use of TS calibration on source. Across all datasets, we observe that ATC achieves superior performance (lower MAE is better). For language datasets, we use DistilBERT-base-uncased, for vision dataset we report results with DenseNet model with the exception of MNIST where we use FCN. We include results on other architectures in App. H.8. For GDE post T and pre T estimates match since TS doesn't alter the argmax prediction. Results reported by aggregating MAE numbers over 4 different seeds.

with other architectures in App. H.8. In App. H.8.1, we also perform ablations on CIFAR using a pre-trained model and observe that pre-training doesn't change the efficacy of ATC.

We predict accuracy on the target data before and after calibration with TS. First, we observe that both ATC-NE and ATC-MC (even without TS) obtain significantly lower MAE when compared with other methods (even with TS). Note that with TS we observe substantial improvements in MAE for all methods. Overall, ATC-NE (with TS) typically achieves the smallest MAE improving by more than 2× on CIFAR and by 3–4× on ImageNet over GDE (the next best alternative to ATC). Alongside, we also observe that a linear fit with robust regression (Siegel, 1982) on the scatter plot recovers a line close to $x = y$ for ATC-NE with TS while the line is far away from $x = y$ for other methods (Fig. 9.2 and Fig. 9.1(right)).

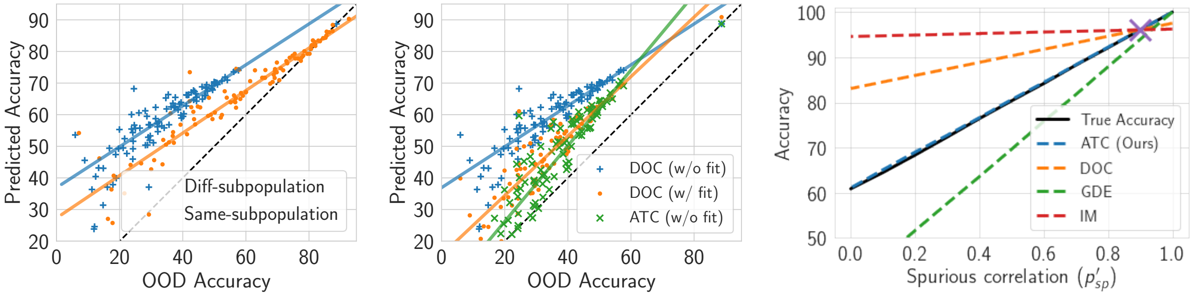


Figure 9.3: **Left:** Predicted accuracy with DOC on Living17 Breeds dataset. We observe a substantial gap in the linear fit of same and different subpopulations highlighting poor correlation. **Middle:** After fitting a robust linear model for DOC on same subpopulation, we show predicted accuracy on different subpopulations with fine-tuned DOC (i.e., DOC (w/ fit)) and compare with ATC without any regression model, i.e., ATC (w/o fit). While observe substantial improvements in MAE from 24.41 with DOC (w/o fit) to 13.26 with DOC (w/ fit), ATC (w/o fit) continues to outperform even DOC (w/ fit) with MAE 10.22. We show parallel results with other Breeds datasets in App. H.8.2. **Right :** Empirical validation of our toy model. We show that ATC perfectly estimates target performance as we vary the degree of spurious correlation in target. '×' represents accuracy on source.

Remarkably, MAE is in the range of 0.4–5.8 with ATC for CIFAR, ImageNet, MNIST, and Wilds. However, MAE is much higher on Breeds benchmark with novel subpopulations. While we observe a small MAE (i.e., comparable to our observations on other datasets) on Breeds with natural and synthetic shifts from the same sub-population, MAE on shifts with novel population is significantly higher with all methods. Note that even on novel populations, ATC continues to dominate all other methods across all datasets in Breeds.

Additionally, for different subpopulations in Breeds setup, we observe a poor linear correlation of the estimated performance with the actual performance as shown in Fig. 9.3 (left)(we notice a similar gap in the linear fit for all other methods). Hence in such a setting, we would expect methods that fine-tune a regression model on labeled target examples from shifts with one subpopulation will perform poorly on shifts with different subpopulations. Corroborating this intuition, next, we show that even after fitting a regression model for DOC on natural and synthetic shifts with source subpopulations, ATC without regression model continues to outperform DOC with regression model on shifts with novel subpopulation.

Fitting a regression model on Breeds with DOC. Using label target data from natural and synthetic shifts for the same subpopulation (same as source), we fit a robust linear regression model (Siegel, 1982) to fine-tune DOC as in Guillory et al. (2021). We then evaluate the fine-tuned DOC (i.e., DOC with linear model) on natural and synthetic shifts from novel subpopulations on Breeds benchmark. Although we observe significant improvements in the performance of fine-tuned DOC when compared with DOC (without any fine-tuning), ATC without any regression model continues to perform better (or similar) to that of fine-tuned DOC on novel subpopulations (Fig. 9.3 (middle)). Refer to App. H.8.2 for details and Table H.2 for MAE on Breeds with regression model.

9.6 Investigating ATC on Toy Model

In this section, we propose and analyze a simple theoretical model that distills empirical phenomena from the previous section and highlights efficacy of ATC. Here, our aim is not to obtain a general model that captures complicated real distributions on high dimensional input space as the images in ImageNet. Instead to further our understanding, we focus on an *easy-to-learn* binary classification task from Nagarajan et al. (2020) with linear classifiers, that is rich enough to exhibit some of the same phenomena as with deep networks on real data distributions.

Consider a easy-to-learn binary classification problem with two features $x = [x_{\text{inv}}, x_{\text{sp}}] \in \mathbb{R}^2$ where x_{inv} is fully predictive invariant feature with a margin $\gamma > 0$ and $x_{\text{sp}} \in \{-1, 1\}$ is a spurious feature (i.e., a feature that is correlated but not predictive of the true label). Conditional on y, the distribution over x_{inv} is given as follows: $x_{\text{inv}}|(y = 1) \sim U[\gamma, c]$ and $x_{\text{inv}}|(y = 0) \sim U[-c, -\gamma]$, where c is a fixed constant greater than γ. For simplicity, we assume that label distribution on source is uniform on $\{-1, 1\}$. x_{sp} is distributed such that $P_s[x_{\text{sp}} \cdot (2y - 1) > 0] = p_{\text{sp}}$, where $p_{\text{sp}} \in (0.5, 1.0)$ controls the degree of spurious correlation. To model distribution shift, we simulate target data with different degree of spurious correlation, i.e., in target distribution $P_t[x_{\text{sp}} \cdot (2y - 1) > 0] = p'_{\text{sp}} \in [0, 1]$. Note that here we do not consider shifts in the label distribution but our result extends to arbitrary shifts in the label distribution as well.

In this setup, we examine linear sigmoid classifiers of the form $f(x) = \left[\frac{1}{1+e^{w^T x}}, \frac{e^{w^T x}}{1+e^{w^T x}}\right]$ where $w = [w_{\text{inv}}, w_{\text{sp}}] \in \mathbb{R}^2$. While there exists a linear classifier with $w = [1, 0]$ that correctly classifies all the points with a margin γ, Nagarajan et al. (2020) demonstrated that a linear classifier will typically have a dependency on the spurious feature, i.e., $w_{\text{sp}} \neq 0$. They show that due to geometric skews, despite having positive dependencies on the invariant feature, a max-margin classifier trained on finite samples relies on the spurious feature. Refer to App. H.4 for more details on these skews. In our work, we show that given a linear classifier that relies on the spurious feature and achieves a non-trivial performance on the source (i.e., $w_{\text{inv}} > 0$), ATC with maximum confidence score function *consistently* estimates the accuracy on the target distribution.

Theorem 9.6.1 (Informal)**.** *Given any classifier with $w_{inv} > 0$ in the above setting, the threshold obtained in* (9.1) *together with ATC as in* (9.2) *with maximum confidence score function obtains a consistent estimate of the target accuracy.*

Consider a classifier that depends positively on the spurious feature (i.e., $w_{\text{sp}} > 0$). Then as the spurious correlation decreases in the target data, the classifier accuracy on the target will drop and vice-versa if the spurious correlation increases on the target data. Theorem 9.6.1 shows that the threshold identified with ATC as in (9.1) remains invariant as the distribution shifts and hence ATC as in (9.2) will correctly estimate the accuracy with shifting distributions. Next, we illustrate Theorem 9.6.1 by simulating the setup empirically. First we pick a arbitrary classifier (which can also be obtained by training on source samples), tune the threshold on hold-out source examples and predict accuracy with different methods as we shift the distribution by varying the degree of spurious correlation.

Empirical validation and comparison with other methods. Fig. 9.3(right) shows that as the degree of spurious correlation varies, our method accurately estimates the target performance where all other methods fail to accurately estimate the target performance. Understandably, due to poor calibration of the sigmoid linear classifier AC, DOC and GDE fail. While in principle IM can perfectly estimate the accuracy on target in this case, we observe that it is highly sensitive to the number bins and choice of histogram binning (i.e., uniform mass or equal width binning). We elaborate more on this in App. H.4.

Biased estimation with ATC. Now we discuss changes in the above setup where ATC yields inconsistent estimates. We assumed that both in source and target $x_{\text{inv}}|y = 1$ is uniform between $[\gamma, c]$ and $x|y = -1$ is uniform between $[-c, -\gamma]$. Shifting the support of target class conditional $p_t(x_{\text{inv}}|y)$ may introduce a bias in ATC estimates, e.g., shrinking the support to $c_1(< c)$ (while maintaining uniform distribution) in the target will lead to an over-estimation of the target performance with ATC. In App. H.4.1, we elaborate on this failure and present a general (but less interpretable) classifier dependent distribution shift condition where ATC is guaranteed to yield consistent estimates.

9.7 Conclusion and future work

In this chapter, we proposed ATC, a simple method for estimating target domain accuracy based on unlabeled target (and labeled source data). ATC achieves remarkably low estimation error on several synthetic and natural shift benchmarks in our experiments. Notably, our work draws inspiration from recent state-of-the-art methods that use softmax confidences below a certain threshold for OOD detection (Hendrycks and Gimpel, 2017; Hendrycks et al., 2019) and takes a step forward in answering questions raised in Deng and Zheng (2021) about the practicality of threshold based methods.

Our distribution shift toy model justifies ATC on an easy-to-learn binary classification task. In our experiments, we also observe that calibration significantly improves estimation with ATC. Since in binary classification, post hoc calibration with TS does not change the effective threshold, in future work, we hope to extend our theoretical model to multi-class classification to understand the efficacy of calibration. Our theory establishes that a classifier's accuracy is not, in general identified, from labeled source and unlabeled target data alone, absent considerable additional constraints on the target conditional $p_t(y|x)$. In light of this finding, we also hope to extend our understanding beyond the simple theoretical toy model to characterize broader sets of conditions under which ATC might be guaranteed to obtain consistent estimates. Finally, we should note that while ATC outperforms previous approaches, it still suffers from large estimation error on datasets with novel populations, e.g., Breeds. We hope that our findings can lay the groundwork for future work for improving accuracy estimation on such datasets.

Chapter 10

Almost Provable Error Bounds Under Distribution Shift via Disagreement Discrepancy

Based on Rosenfeld and Garg (2023): Elan Rosenfeld, and Saurabh Garg. (Almost) Provable Error Bounds Under Distribution Shift via Disagreement Discrepancy. Advances in Neural Information Processing Systems, 2023.

Abstract

In the previous chapter, we investigated heuristic methods for estimating test error under distribution shift. In this chapter, we derive an (almost) guaranteed upper bound on the error of deep neural networks under distribution shift using unlabeled test data. Prior methods either give bounds that are vacuous in practice or give *estimates* that are accurate on average but heavily underestimate error for a sizeable fraction of shifts. Our bound requires a simple, intuitive condition which is well justified by prior empirical works and holds in practice effectively 100% of the time. The bound is inspired by $\mathcal{H}\Delta\mathcal{H}$-divergence but is easier to evaluate and substantially tighter, consistently providing non-vacuous guarantees. Estimating the bound requires optimizing one multiclass classifier to disagree with another, for which some prior works have used sub-optimal proxy losses; we devise a "disagreement loss" which is theoretically justified and performs better in practice. Across a wide range of benchmarks, our method gives valid error bounds while achieving average accuracy comparable to competitive estimation baselines.

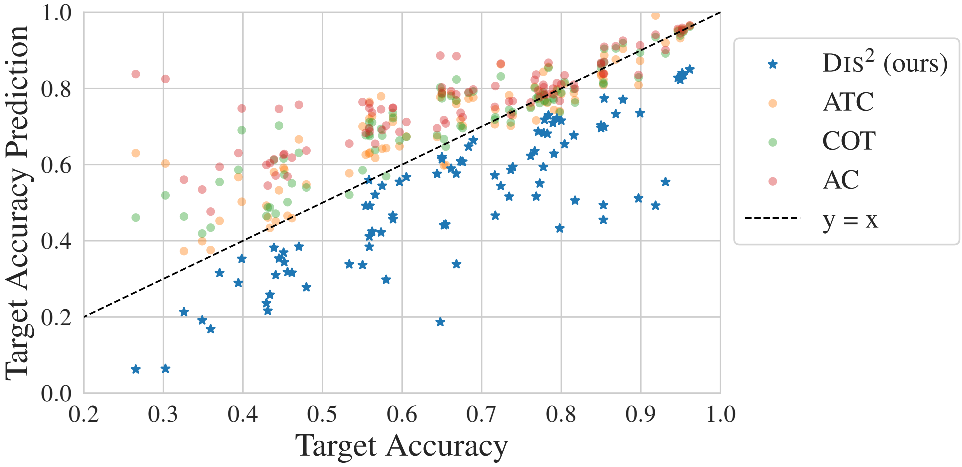


Figure 10.1: *Our bound vs. three prior methods for estimation across a wide variety of shift benchmarks and training methods.* Prior methods are accurate on average, but it is impossible to know if a given prediction is reliable. Worse, they usually overestimate accuracy, with the gap growing as test accuracy decreases—*this is precisely when a reliable, conservative estimate is most desirable.* Instead, DIS^2 maximizes the **dis**agreement **dis**crepancy to give a reliable error bound which holds effectively 100% of the time.

10.1 Introduction

As in previous chapter, to better estimate accuracy in the wild, some recent work instead tries to directly predict accuracy of neural networks using unlabeled data from the test distribution (Baek et al., 2022; Garg et al., 2022c; Lu et al., 2023). While these methods are accurate, they lack pointwise trustworthiness: their estimates are good on average, but they provide no signal of the quality of any individual prediction (here, each point is a *distribution*, for which a method predicts a classifier's accuracy). Indeed, it is reasonably common for them to substantially overestimate test accuracy, which is problematic when optimistic deployment is costly. Worse yet, we find that this gap *grows with test error* (Fig. 10.1), making these predictions least reliable precisely when their reliability is most important. Though it is impossible to give test error upper bounds for all shifts, there is still potential for bounds that are intuitive and reasonably trustworthy.

In this work, we develop a method for (almost) provably bounding test error of classifiers under distribution shift using unlabeled test points. Our bound's only requirement is a simple, intuitive, condition which describes the ability of a hypothesis class to achieve small loss on a objective defined over the (unlabeled) train and test distributions. Inspired by $\mathcal{H}\Delta\mathcal{H}$-divergence (Ben-David et al., 2010b), our method trains a critic to maximize agreement with the classifier of interest on the source distribution while maximizing *dis*agreement on the target distribution; we refer to this joint objective as the *disagreement discrepancy*, and so we name the method DIS^2. We optimize this discrepancy over linear classifiers using deep features—or linear functions thereof—finetuned on the training set. Recent evidence suggests that such representations are sufficient for expressive classifiers even under large distribution shift (Rosenfeld et al., 2022). Experimentally, we find that our bound is valid effectively 100% of the time,[1] consistently giving non-trivial lower bounds on test accuracy which are comparable to competitive baselines. We also show that it is possible to test this bound's likelihood of being satisfied, and we use this to construct a score which can relax the original bound into successively tighter-yet-less-conservative estimates.

While maximizing agreement is statistically well understood, our method also calls for maximizing *dis*agreement on the target distribution. We observe that prior works use losses which do not correspond to minimizing the 0-1 loss of interest and are non-convex (or even *concave*) in the model logits (Chuang et al., 2020; Pagliardini et al., 2023). To rectify this, we derive a new "disagreement loss" which serves as an effective proxy for maximizing multiclass disagreement. Experimentally, we find that minimizing this loss results in higher disagreement.

Experiments across numerous vision datasets demonstrate the effectiveness of our bound. Though DIS^2 is competitive with prior methods for error estimation, we emphasize that our focus is *not* on improving raw predictive accuracy—rather, we hope to obtain reliable (i.e., conservative), reasonably tight bounds on the test error of a given classifier under distribution shift.

10.2 Deriving an (Almost) Provable Error Bound

Notation. Let $\mathcal{S}, \mathcal{T}$ denote the source and target (train and test) distributions, respectively, over labeled inputs $(x, y) \in \mathcal{X} \times \mathcal{Y}$, and let $\widehat{\mathcal{S}}$, $\widehat{\mathcal{T}}$ denote sets of samples from them with cardinalities n_S and n_T (they also denote the corresponding empirical distributions). Recall that we observe only the covariates x without the label y when a sample is drawn from $\mathcal{T}$. We consider classifiers $h : \mathcal{X} \to \mathbb{R}^{|\mathcal{Y}|}$ which output a vector of logits, and we let $\widehat{h}$ denote the particular classifier whose error we aim to bound. Generally, we use $\mathcal{H}$ to denote a hypothesis class of such classifiers. Occasionally, where clear from context, we use $h(x)$ to refer to the argmax logit, i.e. the predicted class. We treat these classifiers as deterministic throughout, though our analysis can easily be extended to probabilistic classifiers and labels. For a distribution $\mathcal{D}$ on $\mathcal{X} \times \mathcal{Y}$, let $\epsilon_{\mathcal{D}}(h, h') := \mathbb{E}_{\mathcal{D}}[\mathbf{1}\{\arg\max_y h(x)_y \neq \arg\max_y h'(x)_y\}]$ denote the one-hot disagreement between classifiers h and h' on $\mathcal{D}$. Let y^* represent the true labeling function such that $y^*(x) = y$ for all samples (x, y); with some abuse of notation, we write $\epsilon_{\mathcal{D}}(h)$ to mean $\epsilon_{\mathcal{D}}(h, y^*)$, i.e. the 0-1 error of classifier h on distribution $\mathcal{D}$.

The bound we derive in this work is extremely simple and relies on one new concept:
Definition 10.2.1. *The* disagreement discrepancy $\Delta(h, h')$ *is the disagreement between h and h' on $\mathcal{T}$ minus their disagreement on $\mathcal{S}$: $\Delta(h, h') := \epsilon_{\mathcal{T}}(h, h') - \epsilon_{\mathcal{S}}(h, h')$.*

We leave the dependence on $\mathcal{S}, \mathcal{T}$ implicit. Note that this term is symmetric and signed—it can be negative. With this definition, we now have the following lemma:

[1] The few violations are expected a priori, have an obvious explanation, and only occur for a specific type of learned representation. We defer a more detailed discussion of this until after we present the bound.

Lemma 10.2.2. *For any h, $\epsilon_{\mathcal{T}}(h) = \epsilon_{\mathcal{S}}(h) + \Delta(h, y^*)$.*

We cannot directly use Theorem 10.2.2 to estimate $\epsilon_{\mathcal{T}}(\widehat{h})$ because the second term is unknown. However, observe that y^* is *fixed*. That is, while a learned $\widehat{h}$ will depend on y^*—and therefore $\Delta(\widehat{h}, y^*)$ may be large under large distribution shift—y^* is *not* chosen to maximize $\Delta(\widehat{h}, y^*)$ in response to the $\widehat{h}$ we have learned. This means that for an expressive hypothesis class $\mathcal{H}$, it should be possible to identify an alternative function $h' \in \mathcal{H}$ for which $\Delta(\widehat{h}, h') \geqslant \Delta(\widehat{h}, y^*)$ (we refer to such h' as the *critic*). In other words, we should be able to find an $h' \in \mathcal{H}$ which, *if it were the true labeling function*, would imply at least as large of a drop in accuracy from train to test as occurs in reality.

In this work we consider the class $\mathcal{H}$ of linear critics, with $\mathcal{X}$ defined as source-finetuned deep neural representations or the resulting logits output by $\widehat{h}$. Prior work provides strong evidence that this class has surprising capacity under distribution shift, including the possibility that functions very similar to y^* lie in $\mathcal{H}$ (Kang et al., 2020; Kirichenko et al., 2022; Rosenfeld et al., 2022). We formalize this intuition with the following assumption:
Assumption 3. *Define $h^* := \arg\max_{h' \in \mathcal{H}} \Delta(\widehat{h}, h')$. We assume $\Delta(\widehat{h}, y^*) \leqslant \Delta(\widehat{h}, h^*)$.*

Note that this statement is guaranteed for $y^* \in \mathcal{H}$; it becomes meaningful when considering restricted $\mathcal{H}$, as we do here. Note also that this assumption is made on a per-classifier basis. This is important because while the above may not hold for every classifier $\widehat{h}$, it need only hold for the classifiers whose error we would hope to bound, which is in practice a very small subset of all classifiers. From Theorem 10.2.2, we immediately have the following result:
Proposition 10.2.3. *Under Assumption 3, $\epsilon_{\mathcal{T}}(\widehat{h}) \leqslant \epsilon_{\mathcal{S}}(\widehat{h}) + \Delta(\widehat{h}, h^*)$.*

Unfortunately, identifying the optimal h^* is intractable, so this bound is still not estimable—we present it as an intermediate result for clarity. To derive the practical bound, we need one more step. In Section 10.3, we derive a "disagreement loss" which we use to maximize the empirical disagreement discrepancy. Relying on this loss, we instead assume:
Assumption 4. *Suppose we identify the critic $h' \in \mathcal{H}$ which maximizes a concave surrogate to the empirical disagreement discrepancy. We assume $\Delta(\widehat{h}, y^*) \leqslant \Delta(\widehat{h}, h')$.*

This is slightly stronger than Assumption 3—the difference in strength between these two assumptions shrinks as the number of available samples grows and as the quality of our surrogate objective improves. Ultimately, our bound holds without these terms, implying that the stronger assumption is reasonable. We can now give our main result:
Theorem 10.2.4 (Main Bound)**.** *Under Assumption 4, with probability $\geqslant 1 - \delta$, $\epsilon_{\mathcal{T}}(\widehat{h}) \leqslant \epsilon_{\widehat{\mathcal{S}}}(\widehat{h}) + \widehat{\Delta}(\widehat{h}, h') + \sqrt{\frac{(n_{\mathcal{S}} + 4n_{\mathcal{T}}) \log 1/\delta}{2n_{\mathcal{S}} n_{\mathcal{T}}}}$.*

The proof is in Appendix I.7. The core message behind Theorem 10.2.4 is that if there is a simple (i.e., linear) critic h' with large discrepancy, the true y^* could plausibly be this function, implying $\widehat{h}$ could have high error—likewise, if no simple y^* could hypothetically result in high error, we should expect low error.

Remark 10.2.1. *Bounding error under distribution shift is impossible without assumptions.*

Prior works which estimate accuracy with unlabeled data rely on experiments, suggesting that whatever condition allows their method to work holds in a variety of settings (Baek et al., 2022; Garg et al., 2022c; Guillory et al., 2021; Lu et al., 2023); using these methods is implicitly *assuming that it will hold for future shifts. Understanding these conditions is thus crucial for assessing whether they can be expected to be satisfied. It is therefore of great practical value that Assumption 4 is simple and intuitive: below we demonstrate that this simplicity allows us to identify potential failure cases* a priori.

Dis^2 vs. $\mathcal{H}\Delta\mathcal{H}$-Divergence One early attempt at bounding error under shift was $\mathcal{H}$*-divergence* (Ben-David et al., 2006; Mansour et al., 2009) which measures the ability of a binary hypothesis class to discriminate between $\mathcal{S}$ and $\mathcal{T}$. This was later refined to $\mathcal{H}\Delta\mathcal{H}$*-divergence* (Ben-David et al., 2010b), which is equal to $\mathcal{H}$-divergence where the discriminator class comprises all xors between pairs from the original class. Though this measure can in principle provide non-vacuous bounds, it usually does not, and evaluating it is intractable. Furthermore, these bounds are too conservative even for simple functions and distribution shifts because they use uniform convergence. In practice, *we do not care* about bounding the error of all classifiers in $\mathcal{H}$—we only care to bound the error of $\widehat{h}$. More importantly, one should not expect the distribution shift to be *truly* worst case, because the $\mathcal{T}$ and y^* are not chosen adversarially with respect to $\widehat{h}$. Fig. I.7 in the appendix gives a simple demonstration of this point, along with a detailed discussion.

A setting where Dis^2 may be invalid. There is one setting where Assumption 4 is less likely to be satisfied: when the representation we are using is regularized to keep $\max_{h' \in \mathcal{H}} \Delta(\widehat{h}, h')$ small. This occurs for domain-adversarial training methods which penalize the ability to discriminate between $\mathcal{S}$ and $\mathcal{T}$ in feature space. It follows that for these methods Dis^2 should not be expected to hold universally, and we observe this in practice (Fig. I.8). Nevertheless, when Dis^2 does overestimate accuracy, it does so by significantly less than prior methods.

10.3 Efficiently Maximizing the Discrepancy

For a classifier $\widehat{h}$, Theorem 10.2.4 clearly prescribes how to bound its error; the difficulty remains in identifying the maximizing $h' \in \mathcal{H}$. We can approximately minimize $\epsilon_{\mathcal{S}}(\widehat{h}, h')$ by minimizing the convex surrogate $\ell_{\log} := -\log \operatorname{softmax}(h(x))_y$ as justified by statistical learning theory, but it is less clear how to maximize $\epsilon_{\mathcal{T}}(\widehat{h}, h')$. A few prior works suggest proxy losses for multiclass disagreement (Chuang et al., 2020; Pagliardini et al., 2023). We observe that these losses are not theoretically justified, as they do not upper bound the 0-1 disagreement loss we hope to minimize and are non-convex (or even concave) in the model logits. Instead, we derive a new loss which satisfies the above desiderata and thus serves as a more principled approach to maximizing disagreement.

Definition 10.3.1. *The* disagreement logistic loss *of a classifier h on a labeled sample (x, y) is defined as* $\ell_{dis}(h(x), y) := \frac{1}{\log 2} \log \left(1 + \exp \left(h(x)_y - \frac{1}{|\mathcal{Y}|-1} \sum_{\hat{y} \neq y} h(x)_{\hat{y}}\right)\right)$.

Fact 10.3.2. *The disagreement logistic loss is convex in $h(x)$ and upper bounds the 0-1 disagreement loss (i.e., $\mathbf{1}\{\arg\max_{\widehat{y}} h(x)_{\widehat{y}} = y\}$). For binary classification, it is equivalent to the logistic loss with the label flipped.*

We expect that ℓ_{dis} can serve as a useful drop-in replacement for any future algorithm which requires maximizing disagreement in a principled manner. We combine ℓ_{log} and ℓ_{dis} to get the empirical disagreement discrepancy objective:

$$\widehat{\mathcal{L}} := \frac{1}{|\widehat{\mathcal{S}}|}\sum_{\widehat{\mathcal{S}}} \ell_{\text{log}}(h'(x), \widehat{h}(x)) + \frac{1}{|\widehat{\mathcal{T}}|}\sum_{\widehat{\mathcal{T}}} \ell_{\text{dis}}(h'(x), \widehat{h}(x)).$$

In practice we optimize this objective with multiple initializations and hyperparameters and select the solution with the largest empirical discrepancy on a holdout set to ensure a conservative bound. Experimentally, we find that replacing ℓ_{dis} with either of the surrogate losses from (Chuang et al., 2020; Pagliardini et al., 2023) results in smaller discrepancy; we present these results in Appendix I.1.

Tightening the bound by optimizing over logits. It is clear that the value of the bound in Theorem 10.2.4 will decrease as $\mathcal{H}$ is restricted. Since the number of features is large, one may expect that Assumption 4 holds even for a reduced feature set. In particular, it is well documented that deep networks experience *neural collapse* (Papyan et al., 2020), giving representations whose effective rank is approximately equal to the number of classes. This suggests that the logits themselves should contain most of the features' information about $\mathcal{S}$ and $\mathcal{T}$. To test this, we evaluate $\textsc{Dis}^2$ on the full features, the logits output by $\widehat{h}$, and various fractions of the top principal components (PCs) of the features. We observe that using logits indeed results in tighter error bounds *while still remaining valid*—in contrast, using fewer top PCs also results in smaller error bounds, but at some point they become invalid (Fig. I.2). The bounds we report in this work are thus evaluated on the logits of $\widehat{h}$, except where we provide explicit comparisons in Section 10.4.

Identifying the ideal number of PCs via a "validity score". Even though reducing the feature dimensionality eventually results in an invalid bound, we may hope to identify approximately when this occurs, giving a more accurate (though less conservative) prediction. We find that *the optimization trajectory itself* provides meaningful signal about this change. We design a "validity score" which captures this information and we observe that it is roughly linearly correlated with the tightness of the bound (Fig. I.4). We can thus evaluate $\textsc{Dis}^2$ with successively fewer PCs and only retain those above a certain score threshold, reducing MAE while remaining reasonably conservative (Fig. I.5). For further details, see Appendix I.2.

10.4 Experiments

Datasets. We conduct experiments across 11 diverse vision benchmark datasets for distribution shift on datasets that span applications in object classification, satellite imagery,

	MAE (↓)		Coverage (↑)		Overest. (↓)	
DA?	✗	✓	✗	✓	✗	✓
Prediction Method						
AC (Guo et al., 2017)	0.1055	0.1077	0.1222	0.0167	0.1178	0.1089
DoC (Guillory et al., 2021)	0.1046	0.1091	0.1667	0.0167	0.1224	0.1104
ATC NE (Garg et al., 2022c)	0.0670	0.0838	0.3000	0.1833	0.0842	0.0999
COT (Lu et al., 2023)	0.0689	0.0812	0.2556	0.1833	0.0851	0.0973
Dis^2 (Features)	0.2807	0.1918	1.0000	1.0000	0.0000	0.0000
Dis^2 (Logits)	0.1504	0.0935	0.9889	0.7500	0.0011	0.0534
Dis^2 (Logits w/o δ)	0.0829	0.0639	0.7556	0.4167	0.0724	0.0888

Table 10.1: **Comparing the Dis^2 bound to prior methods for predicting accuracy.** DA denotes if the representations were learned via a domain-adversarial algorithm. In addition to mean absolute error (MAE), we report what fraction of predictions correctly bound the true error (Coverage), and the average prediction error among shifts whose accuracy is overestimated (Overest.). Dis^2 has reasonably competitive MAE but substantially higher coverage. By dropping the concentration term in Theorem 10.2.4 we can do even better, at some cost to coverage.

and medicine. Because distribution shifts vary widely in scope, prior evaluations which focus on only one specific type of shift (e.g., corruptions) or algorithm often do not convey the full story. We therefore emphasize the need for more comprehensive evaluations across many different types of shifts and training methods, as we present here. We also experiment with Unsupervised Domain Adaptation (UDA) methods which aim to improve target performance with unlabeled target data.

Methods and metrics. We compare Dis^2 to four competitive baselines: *Average Confidence* (AC), *Difference of Confidences* (DoC), *Average Thresholded Confidence* (ATC), and *Confidence Optimal Transport* (COT). For Dis^2, we report bounds evaluated on both full features and logits as described in Section 10.3. Unless specified otherwise, we set $\delta = .01$ everywhere. We also experiment with dropping the lower order term in Theorem 10.2.4. As is standard, we report the *mean absolute error* (MAE)—since our emphasis is on conservative error bounds, we also report the *coverage*, i.e. the fraction of predictions for which the true error does not exceed the predicted error. Finally, we measure the average overestimation: this is the MAE among predictions which overestimate the accuracy.

Results. Table 10.1 reports metrics for all methods. We stratify only by whether the training method is domain-adversarial (DA), as this affects Assumption 4. We find that Dis^2 achieves competitive MAE while maintaining substantially higher coverage, even for DA features. When it does overestimate accuracy, it does so by much less, implying that it is ideal for conservative estimation even when any given error bound is not technically satisfied. We also visualize performance on individual distribution shifts, plotting each source-target pair as a single point for DA (Fig. I.8) and non-DA methods (Fig. 10.1). These plots do not include DoC, as it performed comparably to AC.

Strengthening the baselines to improve coverage. Since the baselines prioritize predictive accuracy over conservative estimates, their coverage might be improvable without too much increase in error. We attempt this with a simple post-hoc adjustment in Appendix I.3. We find that (i) the baselines do not achieve the desired coverage level, though they get somewhat close; and (ii) the adjustment causes them to suffer higher MAE than $\textsc{Dis}^2$. Thus $\textsc{Dis}^2$ is on the Pareto frontier of MAE and coverage, and is preferable when conservative bounds are desirable.

10.5 Conclusion

The ability to evaluate *trustworthy*, non-vacuous error bounds for deep neural networks under distribution shift remains an extremely important open problem. Prior methods which estimate accuracy using extra information—such as unlabeled test samples—rely on opaque conditions whose likelihood of being satisfied is difficult to predict, and so they sometimes provide large overestimates of test accuracy with no warning signs. This chapter attempts to bridge this gap with a simple, intuitive condition and a new disagreement loss which together result in competitive error *prediction*, while simultaneously providing an (almost) provable probabilistic error *bound*. We also study how the process of evaluating the bound can provide even more useful signal. We expect there is potential to push further in each of these directions, hopefully extending the current accuracy-reliability Pareto frontier for test error bounds under distribution shift.

Part IV

Handling Distribution Shifts with Vision Language Models

Chapter 11

TiC-CLIP: Continual Training of CLIP Models

Based on Garg et al. (2024): Saurabh Garg, Mehrdad Farajtabar, Hadi Pouransari, Raviteja Vemulapalli, Sachin Mehta, Oncel Tuzel, Vaishaal Shankar, and Fartash Faghri. Tic-clip: Continual training of clip models. International Conference on Learning Representations (ICLR), 2024.

Abstract

Keeping large foundation models up to date on latest data is inherently expensive. To avoid the prohibitive costs of constantly retraining, it is imperative to *continually* train these models. This problem is exacerbated by the lack of any large scale continual learning benchmarks or baselines. We introduce the first set of web-scale Time-Continual (TiC) benchmarks for training vision-language models: TiC-DataComp, TiC-YFCC, and TiC-RedCaps. TiC-DataComp, our largest dataset, contains over 12.7B timestamped image-text pairs spanning 9 years (2014–2022). We first use our benchmarks to curate various *dynamic* evaluations to measure temporal robustness of existing models. We show OpenAI's CLIP (trained on data up to 2020) loses $\approx$ 8% zero-shot accuracy on our curated retrieval task from 2021–2022 compared with more recently trained models in OpenCLIP repository. We then study how to efficiently train models on time-continuous data. We demonstrate that a simple rehearsal-based approach that continues training from the last checkpoint and replays old data reduces compute by 2.5× when compared to the standard practice of retraining from scratch. Code is available at this url.

11.1 Introduction

In this part of thesis, we switch our focus to distribution shift problems commonly faced with foundations models trained on internet data. Large multimodal foundation models (Bommasani et al., 2021) have offered unprecedented advancements in image-generation

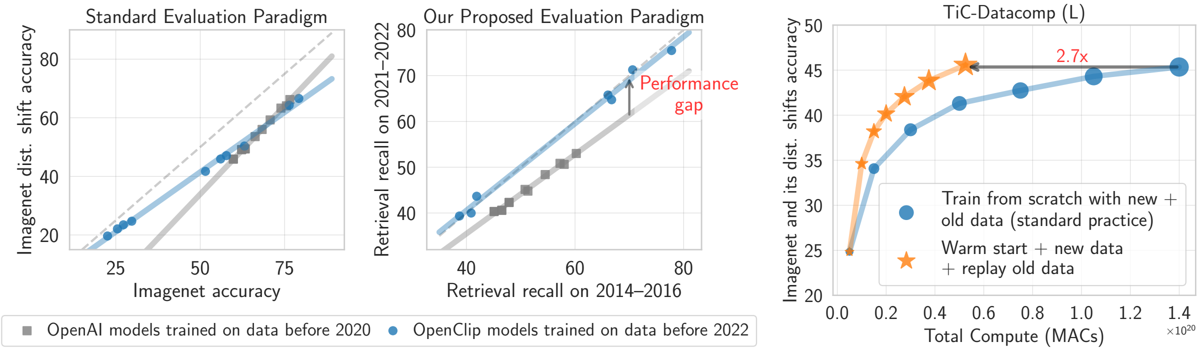


Figure 11.1: *(Left, Middle)* **OpenAI models show less zero-shot robustness on retrieval task from 2021–2022.** OpenCLIP models and OpenAI models have similar robustness on standard benchmarks. However, OpenAI models show less robustness on our retrieval task when compared with recent models in OpenCLIP repository, highlighting susceptibility to a time-evolving data distribution *(Right)* **Simple continual training baseline is computationally efficient and competitive to retraining from scratch.** Different points denote models trained sequentially on our TiC-DataComp (L) as data arrives over time. Warm start training with previous checkpoint and replaying all old data, performs similar to Oracle which trains from scratch every time new data arrives, by using 2.7× less compute.

and zero-shot generalization, and have led to a paradigm shift in multimodal learning, e.g., CLIP (Radford et al., 2021), Flamingo (Alayrac et al., 2022), and Stable Diffusion (Rombach et al., 2022). These foundation models are typically trained on large web-scale datasets which are fixed and *static* in nature. For example, CLIP's training data contains 400 million image-text pairs, and Stable Diffusion was trained on LAION-2B dataset (Schuhmann et al., 2022). In reality, however, these models must operate in a *dynamic* environment, where the world is in a state of constant change. For instance, the internet continually evolves, with petabytes of new data being added daily (Wenzek et al., 2019; Wiener and Bronson, 2014). It remains unclear how legacy models, e.g., OpenAI's CLIP models which were trained on internet-scale data up until 2020, work on future data and whether they even require any re-training to adapt to time-evolving data.

We begin by comparing robustness of OpenAI's CLIP models to others in OpenCLIP repository that are trained on more recently curated web-datasets (e.g., LAION-5B, DataComp) containing data up until 2022 (Ilharco et al., 2021). Since there is no existing benchmark to understand robustness to time-evolving vision-language data, we curate *dynamic* classification and retrieval tasks for years 2014–2022 and evaluate different CLIP models (see Sec. 11.2.2 for our evaluation tasks). We make an intriguing observation that OpenAI models exhibit a significant gap in retrieval performance on data from 2021–2022 compared with 2014–2016 whereas OpenCLIP models retain their performance. In contrast, standard evaluations such as accuracy on ImageNet distribution shifts paint an incomplete picture that OpenAI's CLIP models are slightly more robust than OpenCLIP models (Fig. 11.1). Our findings not only demonstrate the critical need for models to adapt and evolve alongside dynamic data distributions, but also underscores the limitations of relying solely on static benchmarks (e.g. ImageNet).

One naive but common practice for adapting to time-evolving data is to train a new CLIP model from *scratch* every time we obtain a new pool of image-text data. This practice has its rationale: initiating training from a pre-existing model can make it difficult to change the model's behavior in light of new data (Achille et al., 2018; Ash and Adams, 2020; Liu et al., 2023). However, training foundation models from scratch demands significant computational resources and is often infeasible to repeat frequently. For example, ViT-g-14 in Cherti et al. (2022); Schuhmann et al. (2022) was trained for 240K A100 GPU hours which is approximately one month on 400 GPUs. The prevailing training guidelines centered around scaling laws for CLIP training have only looked at training from scratch (Cherti et al., 2023). This leads to a pivotal question: *How can we continuously update models as the data distribution evolves over time given computational constraints?*

There exists a vast literature on continual learning, with a focus on adapting models to dynamic environments (De Lange et al., 2021; Hadsell et al., 2020; Parisi et al., 2019). Traditionally, this field concentrated on synthetic incremental benchmarks that lack natural evolution between tasks, and hence, continual learning methods are seldom used in real-world scenarios (Cossu et al., 2022; Lin et al., 2021). In contrast, recent works focusing on continual learning methods for CLIP models, primarily target improving performance on a single or a sequence of disjoint downstream tasks (Ding et al., 2022; Ilharco et al., 2022; Zheng et al., 2023; Zhou et al., 2023b). While some recent works have started to address these problems, existing benchmarks are comparatively much smaller in scale, or lack paired image-text data (Lin et al., 2021; Ni et al., 2023). Simply put, there is a scarcity of work focusing on continual training of CLIP models on naturally evolving data with time at web-scale.

We take the first step towards **Time-Continual (TIC)** training of CLIP models where data distribution evolves naturally over time (overview in Fig. 11.2). We introduce TIC-DataComp, a new benchmark for Time-Continual training of CLIP models, which we create by appending "crawl time" information to existing CommonPool dataset (Gadre et al., 2023). We also repurpose other web-scale datasets gathered from diverse sources, such as Reddit and Flickr. Specifically, we curate TIC-YFCC and TIC-RedCaps by leveraging time information available in YFCC (Thomee et al., 2016) and Redcaps (Desai et al., 2021) respectively. The primary objective of our study on this benchmark is to develop continual learning methods that operate within a constrained computational budget (say C) each time a fresh batch of data becomes available. These methods compete with an Oracle, which starts training from scratch every time new data arrives, utilizing a cumulative computational budget.

To assess models trained in our TIC-CLIP framework, we evaluate models on our proposed dynamic evaluation tasks that evolve with time along with 28 standard classification and retrieval tasks including ImageNet (Krizhevsky et al., 2012), ImageNet distributions shifts, and Flickr (Plummer et al., 2015), in a zero-shot manner following the work of Gadre et al. (2023); Radford et al. (2021).

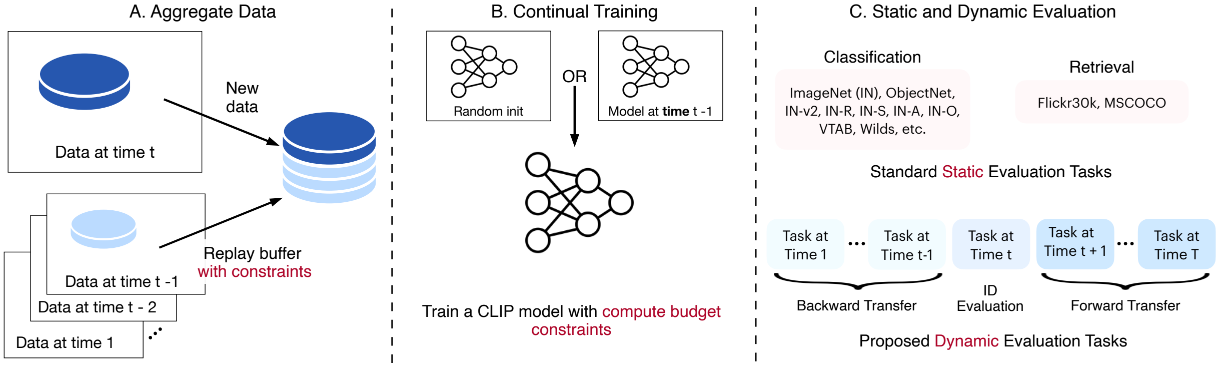


Figure 11.2: **Experimental protocol on our proposed continual benchmarks.** *(A)* Combine new and old data given buffer constraints. *(B)* Continually train a model with a compute budget (say C) either by starting with previous checkpoint or from scratch. *(C)* Evaluate models on standard datasets and our proposed dynamic datasets. Comparison with other benchmarks in Appendix J.1.

Finally, we develop continual learning methods on our benchmarks and perform over two hundred experiments with different baselines that utilize previous checkpoints (e.g., warm start, patching, and distillation), replay buffers, and learning rate schedules. Our findings highlight a key takeaway: Cumulative method that warm starts training with the latest checkpoint and replays all old data, achieves performance competitive to an Oracle while being 2.7× computationally more efficient. Additionally, our experiments demonstrate interesting trade-offs between buffer sizes for static and dynamic performance and provide valuable insights into learning rate schedules for sequential training. Our results span over various dataset scales (from 11M samples to 3B) and highlight trends with different methods that are largely consistent across scales.

To make our benchmarks accessible, we publicly release the code and the time information we collect on top of existing datasets here. Our work is just an initial step towards continual training of foundation models, and we believe our research would spur more attention to this understudied area.

11.2 TiC-CLIP: Benchmarks and Experimental Protocol

In this section, we introduce our benchmark (Fig. 11.2) focusing on the training of a vision-language foundation model with the Contrastive Language Image Pretraining (CLIP) (Radford et al., 2021)) objective. Notably, we train on image-text data that arrives sequentially unlike the conventional image-text datasets which are static (e.g. WiT in CLIP, DataComp in Gadre et al. (2023)). We curate TiC-DataComp, TiC-YFCC, and TiC-RedCaps that are image-text pairs sourced from the internet which we augment with auxiliary time information. We also introduce dynamic evaluation tasks to assess performance of our continually trained models on data evolving with time. The goal of a learner is to train a *deployable* model at each step as new data becomes available with a fixed compute budget.

11.2.1 Benchmark Design: How we Create Time-Continual Datasets?

To instantiate continual training of CLIP, we extend existing image-text datasets with time information collected from the original source of the datasets. Our largest dataset is TiC-DataComp which contains 12.7 billion image-text pairs with "crawl-time" metadata. We create this dataset on top of the existing DataComp benchmark (Gadre et al., 2023). We also create TiC-YFCC and TiC-RedCaps on top of existing YFCC15M (Radford et al., 2021; Thomee et al., 2016) and Redcaps (Desai et al., 2021) datasets to highlight that our findings are broadly applicable to carefully curated datasets from diverse sources such as Reddit and Flickr. While time-related metadata is absent in the DataComp benchmark, it is available in the original releases of YFCC and Redcaps. Nevertheless, to the best of our knowledge, no prior work utilizes such time information for continual training of CLIP models. We show dataset statistics for all datasets, e.g., number of examples in each year in App. J.3.3.

TiC-DataComp We collect timestamps for the CommonPool dataset introduced in DataComp which contains 12.7B image-text pairs (not including 0.1B inaccessible ones). This dataset stands as the largest public image-text dataset to date. The source of DataComp is Common Crawl, which periodically releases web-crawled data snapshots, typically on a monthly basis since 2014 with new and updated webpages. To construct TiC-DataComp, we augment each image-text pair in DataComp with their *first* timestamp. We followed the same construction process as DataComp but retained only the image-text pair found in the earliest snapshot during the deduplication stage. This process provides timestamps at the granularity of months, spanning years 2014–2022. See App. J.3.7 for details on the construction process. We note that while this augmented time information may contain some noise, on average, we find it to be a reasonably accurate proxy for the upload time of web pages (see App. J.3.7).

Although our benchmark contains time information at the granularity of months, we limit our experiments to granularity of years by consolidating data for all months in a year. Similar to DataComp, our benchmark has an inclusive design, accommodating participants with varying levels of computational resources. In particular, we experiment with `medium`, `large`, and `xlarge` sizes from CommonPool. (Gadre et al., 2023) leverage different filtering strategies to select the training subset. We are concerned that filtering techniques bias the selected training data. In App. J.3.1, we provide preliminary evidence that "Bestpool" filtering that uses off-the-shelf CLIP models, indeed biases the selected data to old time steps. Nevertheless, to highlight significance of our findings even for state-of-the filtering techniques, we experiment with both Bestpool and Basic filtering (no CLIP filtering) at `xlarge` scale. For `large` and `medium` scales, we only experiment with Basic filtering.

TiC-YFCC We experiment with the 15M subset of YFCC100M (Thomee et al., 2016), namely YFCC15M, selected by OpenAI (Radford et al., 2021). This filtering retains only

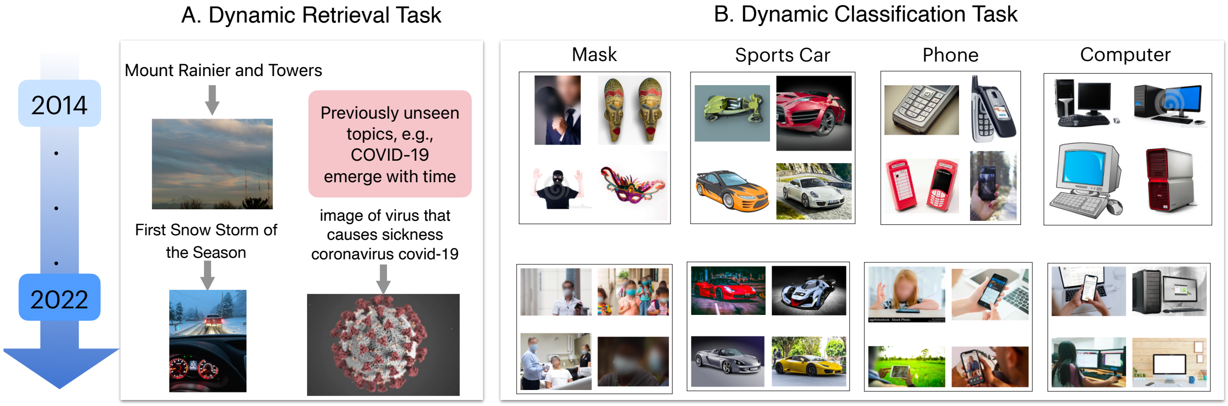

A. Dynamic Retrieval Task
2014
2022
Mount Rainier and Towers
First Snow Storm of the Season
Previously unseen topics, e.g., COVID-19 emerge with time
image of virus that causes sickness coronavirus covid-19
B. Dynamic Classification Task
Mask
Sports Car
Phone
Computer

Figure 11.3: **Distribution of examples changes from 2014 to 2022 in our dynamic evaluation tasks.** *(Left)* Samples for text to image retrieval. For new timestamps, images from novel concepts appear (e.g., COVID-19). *(Right)* Samples from our classification task for 4 categories. We observe that not only objects evolve over time but also images from recent timestamps are captured more in the wild.

images with natural text in captions. YFCC100M contains data from years 2008–2014 and was originally released with upload timestamps. We use this information to create continual splits at the granularity of years.

TIC-RedCaps RedCaps contains 12M image-caption pairs from manually curated set of subreddits across 2011–2020 (Desai et al., 2021). We use the creation timestamps of the posts to create splits for continual learning. Similar to the other two datasets, we experiment at the granularity of years.

11.2.2 Evaluation Testbed

Dynamic tasks We leverage the temporal information in our benchmarks to create *dynamic evaluation* tasks. Here, the test data comprises samples varying over years as the world evolved. For our largest dataset which is TIC-DataComp, we create dynamic tasks for both retrieval and classification as described below. (examples in Figure 11.3 and additional examples in App. J.3.5):

I. *Dynamic retrieval task*: To create a retrieval task, we sample a batch of IID image-text pairs from different timestamps and evaluate text retrieval performance given the corresponding image (similarly, image retrieval given the corresponding text). We refer to the dataset as TIC-DataComp-Retrieval.

II. *Dynamic classification task*: We also create a classification dataset TIC-DataComp-Net with ImageNet classes from CommonPool and augmented with timestamps. Inspired by LAIONNet (Shirali and Hardt, 2023), we first filter examples where the corresponding caption contains one and only one of the synsets of ImageNet. Then we only retain examples where the similarity between ImageNet synset definition and the caption exceeds a threshold of 0.5. We evaluate the similarity using an off-the-shelf sentence embedding model (Reimers and Gurevych, 2019). Crucially, unlike LAIONNet, we do not filter the image-text pairs with

CLIP similarity scores to avoid biasing the selection process. We describe the construction process in more details in App. J.3.5. On TiC-DataComp-Net, we report average accuracy over all classes and over selected nodes (e.g., motor vehicles) at each time step.

Similarly, we create retrieval tasks for TiC-YFCC and TiC-RedCaps. Note that we remove the extracted image-text pairs for dynamic retrieval and classification tasks from the training sets. Evaluations on dynamic tasks are done in a zero shot manner.

Static tasks We also evaluate models on numerous classification and retrieval tasks in a zero-shot manner as in Radford et al. (2021). In particular, we consider 28 standard tasks: 27 image classification tasks, e.g., ImageNet and its 6 distribution shifts (e.g., ImageNetv2, ImageNet-R, ImageNet-Sketch, and Objectnet), datasets from VTAB and Flickr30k retrieval task. We refer to these as *static evaluation* tasks. We list all the datasets in App. J.3.2.

Evaluation metrics We define metrics for classification tasks and retrieval tasks based on *accuracy* and *Recall@1*, respectively. Let T represent the number of time steps for which we have data. For each training method, we generate a total of T models, each corresponding to the end of training at a particular time step. For static datasets (e.g., ImageNet), we report average performance of T models. However, when dealing with dynamic evaluation datasets, we assess the performance of each of the T models on evaluation datasets collected at all time steps. Consequently, for each model and a dynamic evaluation task, we obtain T performance values. We represent these values using the performance matrix $\mathcal{E}$, where each entry $\mathcal{E}_{i,j}$ signifies the performance of the model obtained after observing training data at time step i when evaluated on a dataset from time step j. The performance matrix $\mathcal{E}$ can also be succinctly summarized using three standard metrics commonly employed in continual learning evaluations (Díaz-Rodríguez et al., 2018; Lin et al., 2021):

- *In-domain performance*: average performance at each training time step (i.e., the diagonal of $\mathcal{E}$)
- *Backward transfer*: average on time steps before each training step (i.e., the lower triangular of $\mathcal{E}$)
- *Forward transfer*: average on time steps following each training step (i.e., the upper triangular of $\mathcal{E}$)

Sometimes, the metrics described above can cause the backward transfer metric to be influenced by later evaluation time steps, biasing the backward transfer metric (refer to App. J.6 for details). Therefore, in App. J.6, we present results using revised metrics that mitigate this issue.

While the static tasks capture performance on standard benchmarks, dynamic tasks capture problems due to distribution shift (for forward transfer) and forgetting (for backward transfer). The goal in our benchmark is to develop continual learning methods that maximize performance on static tasks while simultaneously optimizing for performance on dynamic tasks.

11.2.3 Experimental Protocol For Training

Streaming protocol We follow a streaming protocol, where data is progressively revealed to the learner in large batches with the objective of achieving a deployable model as early as possible after each batch arrives. We conduct experiments with data streaming at the granularity of years and our benchmark supports future research at the granularity of months. Additionally, as the amount of data from earlier time steps is limited (see App. J.3.3), we aggregate data from the earlier time steps into a single larger batch and timestamp it by the latest year in the range. After this aggregation, we have 7 time steps for TiC-DataComp (2016–2022) and 4 for both TiC-YFCC (2011–2014) and TiC-RedCaps (2017–2020). While the number of image-text pairs revealed at each time step are of similar orders of magnitude, the exact number does vary across steps and we do not artificially alter the sizes.

Memory budget We allow methods to use the last model checkpoint at each step as the cost of keeping one checkpoint per month is often negligible. In contrast, the cost of retaining old data can be high and might not be permitted due to data expiration policies. Thus, along with studying methods that retain all old data, we also explore strategies that restrict data persistence (see Sec. 11.3 for details).

Compute budget To ensure a fair comparison between methods, we establish a consistent total compute budget, quantified in terms of Multiply-Accumulate Operations (MACs), and allocate it evenly for training at every time step. Unless specified otherwise, for all methods except Oracle and LwF, we use the same compute budget. For experiments on TiC-DataComp, we refer to compute configurations in DataComp for overall compute. For TiC-RedCaps and TiC-YFCC, we use compute of order `medium` scale in TiC-DataComp. Compute budget details are in App. J.3.4.

11.2.4 Analyzing Distribution Shifts in the Constructed Benchmarks

TiC-DataComp analysis through the lens of constructed evaluation tasks First, we qualitatively analyze the examples in our retrieval and classification dataset (Fig. 11.3). We observe that over time, in the retrieval task, new concepts like COVID-19 emerge. Likewise, certain ImageNet classes evolve, such as the shift from "masquerad" masks to "surgical/protective" masks in their definitions. Moreover, as time evolves, we observe that image quality improves and more images tend to appear in the wild in contrast to centered white background images. Next, we compare performance of OpenAI and OpenCLIP models on our datasets. Here, we only present the main findings, and delegate a detailed discussion to App. J.3.6. We observe a significant performance gap between OpenAI and OpenCLIP models on our dynamic retrieval task (Fig. 11.1). This gap widens notably on retrieval queries where captions mention COVID-19. On the other hand, OpenAI and OpenCLIP models exhibit similar robustness for retrieval on data coming from Flickr highlighting that data from some domains do not exhibit shifts that cause performance drops. For our classification task, we observe a very small drop ($\approx 1\%$) when averaged across all categories. However, we observe a substantial gap on specific subtrees in ImageNet. For example, classes in "motor vehicle" subtree show an approximate 4% performance drop, when comparing OpenAI and OpenCLIP models. These findings highlight that while overall ImageNet classes may remain timeless, certain categories tend to evolve faster than others. Our qualitative and quantitative analysis on TiC-DataComp clearly highlights evolution of distributions and captures different properties than standard benchmarks.

Quantitative analysis on TiC-YFCC We analyze TiC-YFCC using off-the-shelf sentence and image encoders. We first embed images from different time steps with an OpenAI CLIP encoder and then compute Frechet Inception Distance (FID; Seitzer (2020)). As time progresses, we observe that FID distance increases with respect to data from first time step (Fig. J.14 in App. J.3.6). Similarly, we use pretrained sentence transformer to extract top-5 categories from Wordnet Nouns for each caption. We observe that the TV distance over distribution of WordNet Nouns evolves over time when compared to data from the first time step. More details in App. J.3.6.

11.3 TiC-CLIP: How to Continually Train CLIP Models?

In this section, we lay out different methods specifically focus on the following questions (Table 11.1): (i) How to utilize/replay data from previous time steps; (ii) How to leverage previously trained model checkpoints? (iii) What should be the training/optimization procedure?

Data replay methods initialized from the last checkpoint demonstrate strong performance on standard continual learning benchmarks (**??**). We consider replay methods with/without initialization from last checkpoint(s):

Table 11.1: Table summarizing our methods. D: data size in each step, T total time steps, t: current time step, C: compute budget (iterations).

Method	**Each Step**			**Total**
	Train Size	**Init.**	**Compute**	**Compute**
Cumulative-All	tD	Last	C	TC
Cumulative-Exp	$2D$	Last	C	TC
Cumulative-Equal	$2D$	Last	C	TC
Sequential	D	Last	C	TC
Restart	tD	Rand	C	TC
Patching	D	Last Patch	C	TC
LwF	D	Last	$1.2 \times C$	$1.2 \times TC$
Oracle**	tD	Rand	tC	$\frac{(T+1)T}{2}C$

I. **Oracle**: Train a CLIP model from scratch (i.e., random initialization) on all image-text data received till time t using a large compute budget of $t \times C$. Oracle represents a *prohibitively expensive* method that is the most common practice in training large-scale foundation models. The goal of other methods is to perform as close as possible to the Oracle within their limited budget.

II. **Cumulative**: Train each model initialized from last checkpoint on the union of all data up to t with compute budget C. This method is analogous to experience replay (Hayes et al., 2019; Robins, 1995) but with substantially larger buffers than common in the continual learning literature. Given a fixed buffer size for each past step, we observe minimal to no difference between random subsampling and other strategies. After sampling the replay data, we randomly shuffle it together with new data for training. We consider the following strategies for sampling buffer sizes per step:

- **-All**: Replay all previous data.

- **-Exp**: Replay a buffer of size D and reduce the amount of old data by half at each step. For example, at 3-rd time step, we retain $D/2, D/2$ of old data and at 4-th, we retain $D/4, D/4, D/2$ of old data. Along with D data from current step, this method trains on at most $2D$ data in each step.

- **-Equal**: Replay a buffer of size D but split the buffer equally among all previous years. For example, at 4-th step, we retain $D/3, D/3, D/3$ of old data. Along with D data from current time step, this method trains on at most $2D$ data in each step.

III. **Sequential**: Train *only* on the new data starting from the best checkpoint of the previous time step. Sequential is similar to Cumulative but without any replay buffer.

IV. **Restart**: Train each model from scratch (i.e., random initialization) on all the data till time t for compute budget C. Restart is similar to the Oracle but with compute budget C at each time step and similar to Sequential but with random initialization. As such, Restart helps us understand the *forward transfer* and *loss of plasticity* in our benchmark (Ash and Adams, 2020; Dohare et al., 2023).

V. **Patching**: We use sequential patching from Ilharco et al. (2022). Initialize from a patched model of last step and train only on the new data. To obtain a patched model at each time step, we apply weight interpolation with the patched model (if any) trained at time step $t-1$ and the model trained at time step t. We tune the mixing coefficients by optimizing average retrieval performance on previous tasks.

VI. **LwF**: Train only on the new data with a KL divergence penalty between the image-text similarity matrix of last checkpoint and current model on each batch (Ding et al., 2022; Li and Hoiem, 2017). See App. J.5 for results with other continual learning methods, e.g., EWC (Kirkpatrick et al., 2017).

Learning rate schedule The defacto Learning Rate (LR) schedule for training CLIP models is an initial linear increase to a maximum value, i.e., warm up, followed by a cosine decay (Gadre et al., 2023; Radford et al., 2021). We default to using a cosine LR schedule for each sequential run, resulting in a cyclic schedule and observe a significant increase in training loss early in subsequent runs when the LR is high. However, as training progresses, we observe that the increased loss decreases at a faster rate (when compared to training from scratch) allowing us to train with cyclic schedules. We discuss this more and explore an alternate learning rate schedule in App. J.2.5.

Other Training details and hyperparameters Unless specified otherwise, we closely follow the original CLIP training recipe (Radford et al., 2021). We train the CLIP variant with ViT-B/16 as the image encoder (Dosovitskiy et al., 2020). All training and hyperparameters can be found in App. J.4.2.

Table 11.2: **Zero shot performance on our time-continual benchmarks.** * and ** denote methods that violate the compute budget. For static tasks, we tabulate accuracy of the models obtained on the final timestamp. For dynamic tasks, we tabulate forward/backward transfer and ID performance on retrieval tasks (Sec. 11.2.3). For TiC-DataComp (XL), we include results with Bestpool filtering (basic filtering in Table J.2). For all metrics, higher is better.

Benchmark	**Method**	**Compute** (MACs)	**Static Tasks**				**Dynamic Retrieval Tasks**		
			ImageNet	ImageNet dist. shift	Flickr30k	Average over 28 datasets	Backward Transfer	ID Performance	Forward Transfer
TiC-YFCC	Restart	3.4×10^{18}	5.2	3.6	3.0	12.9	13.2	41.4	18.6
	Sequential	3.4×10^{18}	17.3	10.5	15.9	21.9	42.2	48.4	23.7
	Patching	3.4×10^{18}	18.9	11.3	18.5	23.3	44.7	53.4	24.5
	Cumulative-Exp	3.4×10^{18}	24.1	14.3	20.4	25.9	60.4	60.1	27.1
	Cumulative-Equal	3.4×10^{18}	23.9	13.8	20.5	26.3	60.4	60.4	27.1
	Cumulative-All	3.4×10^{18}	**29.3**	**17.6**	**26.8**	**29.6**	**66.4**	**60.2**	**27.6**
	LwF*	4.1×10^{18}	16.9	9.8	14.7	21.2	36.6	56.0	23.2
	Cumulative-All*	3.6×10^{18}	**29.2**	**17.5**	**27.4**	**29.3**	**66.8**	**60.3**	**27.6**
	Oracle**	8.5×10^{18}	**29.2**	**17.0**	**25.9**	**29.0**	**66.1**	**61.8**	**26.9**
TiC-RedCaps	Restart	3.4×10^{18}	11.7	8.5	3.7	18.4	21.3	25.4	22.4
	Sequential	3.4×10^{18}	19.3	13.7	6.2	25.8	33.0	33.6	27.5
	Patching	3.4×10^{18}	21.3	15.2	7.7	26.8	34.8	34.8	27.8
	Cumulative-Exp	3.4×10^{18}	27.3	19.1	10.5	30.0	44.5	42.0	32.6
	Cumulative-Equal	3.4×10^{18}	27.8	19.4	10.0	30.5	44.4	42.0	32.6
	Cumulative-All	3.4×10^{18}	**32.2**	18.7	14.5	31.7	**48.9**	**43.2**	**33.4**
	LwF*	4.1×10^{18}	21.6	14.8	8.2	27.3	35.4	36.0	28.4
	Cumulative-All*	3.6×10^{18}	**32.9**	**23.7**	**14.1**	**32.9**	**49.0**	**43.4**	**33.4**
	Oracle**	8.5×10^{18}	**32.7**	**22.7**	**14.3**	**32.3**	**48.5**	**43.1**	**33.4**
TiC-DataComp (M)	Sequential	3.0×10^{18}	19.2	16.4	16.4	26.0	25.7	26.4	14.9
	Patching	3.0×10^{18}	19.3	16.8	18.5	26.4	26.9	25.4	14.5
	Cumulative-Exp	3.0×10^{18}	22.1	18.4	20.4	28.8	31.7	27.1	**15.2**
	Cumulative-Equal	3.0×10^{18}	22.1	18.4	19.2	28.0	31.8	26.8	15.1
	Cumulative-All	3.0×10^{18}	24.0	20.2	20.9	30.0	33.8	26.4	15.1
	LwF*	3.8×10^{18}	19.2	16.5	17.7	27.0	25.6	26.6	14.9
	Cumulative-All*	3.9×10^{18}	**30.0**	**25.0**	**28.6**	**35.1**	**36.7**	**28.3**	**15.5**
	Oracle**	1.2×10^{19}	25.5	21.2	23.3	30.8	34.9	27.8	**15.6**
TiC-DataComp (L)	Sequential	2.7×10^{19}	44.7	37.4	48.4	45.7	52.6	**58.4**	41.1
	Patching	2.7×10^{19}	45.8	38.9	49.7	46.9	55.2	57.5	40.9
	Cumulative-Exp	2.7×10^{19}	47.3	39.6	50.8	47.6	60.4	**58.4**	**41.4**
	Cumulative-Equal	2.7×10^{19}	47.7	40.3	51.8	47.7	60.9	**58.2**	**41.4**
	Cumulative-All	2.7×10^{19}	48.9	41.3	50.9	48.0	62.1	57.3	41.2
	Cumulative-All*	4.1×10^{19}	53.0	**44.3**	**54.4**	**51.3**	63.0	57.8	41.2
	Oracle**	1.1×10^{20}	**53.6**	44.0	53.9	50.4	**64.3**	**58.6**	**41.8**
TiC-DataComp (XL)	Sequential	2.7×10^{20}	66.5	54.2	61.2	61.0	63.1	68.9	56.8
	Cumulative-All	2.7×10^{20}	71.6	58.8	65.1	64.8	**70.7**	**68.5**	**57.1**
	Cumulative-All*	3.5×10^{20}	**72.8**	60.4	66.5	**66.7**	**71.0**	**68.6**	**57.1**
	Oracle**	1.1×10^{21}	**73.3**	**61.3**	**68.0**	65.8	-	-	-

11.4 Experiments and Main Results

Our main results are in Table 11.2 and more detailed plots on each dataset are in App. J.2.1. Recall, our goal is compete with an Oracle that re-trains from scratch every time new data is observed, both on dynamic and static tasks, while being computationally efficient. Here, we summarize our key findings:

Cumulative-All saves up to 4× the cost. On dynamic evaluation tasks, we observe that Cumulative-All where we replay all the past data, achieves performance close to the Oracle (within 1%) using significantly less compute (4× less on TiC-DataComp and 2.5× less on TiC-YFCC and TiC-RedCaps). On static tasks, the gap remains small at small scales but grows to 4.7% on `large`, 1.8% on `xlarge` Bestpool, and 4% on `xlarge` Basic (see Table 11.2 and Table J.2). In these cases, training Cumulative models with slightly

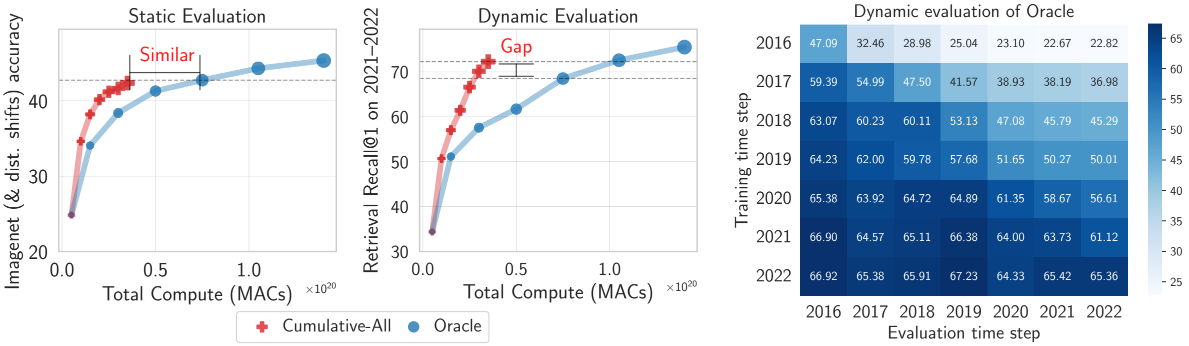


Figure 11.4: *(Left)* **Dynamic and static evaluations rank models differently**. Models with similar performance on static datasets, have $> 6\%$ difference on retrieval task from 2021-2022 TIC-DataComp (L). Different points denote models trained sequentially over time. *(Right)* **Performance of Oracle on future time steps drops highlighting distribution shift in dataset**. Each row evaluates the Oracle trained on TIC-DataComp (L) at a particular time step across all dynamic retrieval tasks.

extra compute bridges the gap while remaining at least 2.7× more computationally efficient (see rows with * in Table 11.2). This highlights that with unconstrained access to past data, we can simply train sequentially and save significant computational resources.

At scale, Sequential has strong forward transfer but lacks on static tasks. On TIC-YFCC and TIC-RedCaps, which are at the smallest scale, we observe a significant gap ($> 10\%$) between Sequential (with no data replay) and Oracle on all tasks. On the other hand, on all scales in TIC-DataComp, Sequential shows strong performance on forward transfer and ID dynamic evaluations. However, on static tasks and backward transfer evaluations, Sequential significantly underperforms the Oracle.

Patching and LwF improve over Sequential but lag behind Cumulative-All. On static tasks, LwF improves over Sequential by 2%, while on dynamic tasks, LwF improves backward transfer by 7% on TIC-DataComp (M). However, its computation cost is higher than even Cumulative-All* which outperforms LwF on all tasks. Patching improves over Sequential on backward transfer on all datasets (e.g., 5% boost on TIC-DataComp L) highlighting that Patching combines benefits of previously patched model and the new Sequential model without additional computation cost. However, such benefits do not show up on static tasks. These results hint that to continuously improve on static tasks with time, replaying old data as in Cumulative-All plays a crucial role.

-Exp and -Equal significantly reduce replay buffer size and maintain static task performance and backward transfer. Recall, that -Exp and -Equal reduce the replay buffer size to a maximum $2D$ of old data. In particular, at the last time step, -Exp and -Equal reduce the buffer size by 3.5× for TIC-DataComp datasets. While reducing the buffer sizes, these methods still achieve performance close to Cumulative-All (within 2%) on both static and dynamic tasks, with -Equal consistently better than -Exp strategy. As we go to large scale, e.g., from `medium` to `large`, the gap between these methods and

Cumulative-All reduces. These findings demonstrate that even a small amount of replay data from old time steps stays competitive with replaying all data and significantly improves over no replay at all.

Warm up helps training on data from first time step, but hurts on subsequent time steps. Cosine LR is commonly coupled with an initial warm-up that linearly increases the LR from zero to maximum LR. We investigate the effectiveness of warm-up in first versus subsequent time steps. Surprisingly, we observe that not using warmup for subsequent training runs is *strictly* more beneficial than using warm up on both static and dynamic tasks. In particular, on TiC-DataComp (L), we observe about 1.5% improvement in ImageNet accuracy and 4.3% improvement on ID dynamic retrieval when not using warmup with Cumulative (see App. J.2.3). Moreover, we also ablate over not using warm up for the first training run and observe a drop of approximately 4.8% accuracy in the first time step on TiC-DataComp (L). Hence, we default to using warmup when training on the first time step and not using it on the subsequent time steps with all methods except for training on TiC-DataComp (XL) where we add a smaller warm up (10% of the warm up iterations used in first step) to stabilize training.

Same maximum LR works best across all runs when using cosine schedule. We ablate on TiC-DataComp (M) to investigate how to change LR after training on data from the first time step. Unlike conventional pretraining and finetuning settings where LR is typically decreased for subsequent training, we observe that decaying maximum LR for subsequent steps in our setup hurts on static and dynamic tasks and consequently, we use same maximum LR across our runs (see App. J.2.3).

Filtering strategy changes the ordering of performance on static and dynamic retrieval tasks. We observe that while bestpool filtering models outperform basic filterining models on TiC-DataComp (XL) by 6% on static tasks, they underperform by over 5% on dynamic retrieval task (see Fig. J.3).

Dynamic tasks provide complimentary information for model selection compared to static tasks. Choosing models solely based on static task performance may inadvertently select models that underperform on dynamic tasks. For example, Cumulative models that show relatively modest improvements on static tasks continue to improve by > 6% for retrieval on 2021-2022 (Fig. 11.4).

Table 11.3: ImageNet continual training. Cumulative-All remains close to Oracle.

Method	**Number of splits**			
	1 (Oracle)	2	4	8
Cumulative-All	80.9	80.8	80.6	80.0

Cumulative-All remains competitive to Oracle even on ImageNet on up to 8 splits. CLIP models are often trained for fewer epochs and are typically not trained until they reach an "overfitting" regime. Here, we investigate how Cumulative-All performs when compared to Oracle when training is done for longer. Specifically, we assess Cumulative-All on 2, 4 and 8 IID splits including the full dataset (see App. J.4.1 for details). Table 11.3 summmarizes our key findings. Notably, even with up to 8 splits, the difference in accuracy between Oracle and Cumulative-All remains below

0.9%. These results underscore the feasibility of continual training with Cumulative-All even on ImageNet.

11.5 Related Work

Benchmarks for continual learning Traditionally, the continual learning community has focused on domain, class, and task incremental benchmarks (Hsu et al., 2018; Van de Ven and Tolias, 2019; Zhou et al., 2023a) with artificial task boundaries (e.g., Split-CIFAR, Perm-MNIST). These benchmarks are often task-specific and present minimal or no meaningful evolution between adjacent tasks. Consequently, continual learning methods are often confined to these benchmarks and seldom scale to practical real-world scenarios (Cossu et al., 2022; Lin et al., 2021). On the other hand, continual learning methods for CLIP models are primarily aimed at fine-tuning to improve performance on a single or on a sequence of disjoint downstream tasks (Ilharco et al., 2022; Thengane et al., 2022; Zheng et al., 2023). Existing large-scale benchmarks for training CLIP models, e.g., Datacomp (Gadre et al., 2023) and LAION-5B (Schuhmann et al., 2022), are curated to investigate methods and scaling laws to train state-of-the-art CLIP models in a single training run. In our work, we augment these existing datasets with temporal information to create benchmarks for continual pertaining of CLIP models.

Continual learning methods Common methods can be categorized into three categories: i) regularization, ii) replay, and iii) architecture-based methods. Regularization methods add a penalty to keep the fine-tuned model close to its initialization and often incur additional memory/compute costs (Farajtabar et al., 2020; Kirkpatrick et al., 2017; Mirzadeh et al., 2020a;b). Data replay methods retain all or a subset of the prior data for subsequent training (Chaudhry et al., 2018; Lopez-Paz and Ranzato, 2017; Rebuffi et al., 2017). Simple replay-based baselines surpass various methods on standard benchmarks (Balaji et al., 2020; Lomonaco et al., 2022; Prabhu et al., 2020). Lastly, architecture-based methods expand the model as new tasks arrive, limiting their applicability in evolving environments without clear task boundaries (Rusu et al., 2016; Schwarz et al., 2018). In this work, we compare popular continual learning methods with simple alternatives for continually pretraining of CLIP.

11.6 Conclusion and Future Work

We view TiC-DataComp as the initial stride toward the continual training of large-scale vision-language foundation models. We believe that our benchmark, alongside the preliminary results obtained using simple baselines will foster future research for large-scale continual-learning. There are several pivotal directions for future work: (i) Compare our baselines on continually streaming data at finer granularity, e.g., streaming data at the monthly level; (ii) Investigate alternate learning rate schedules (e.g., Const-Cosine as in App. J.2.5) that are forward looking, and are better suited to continual learning; (iii) Better data filtering techniques that are more inclusive of future data;

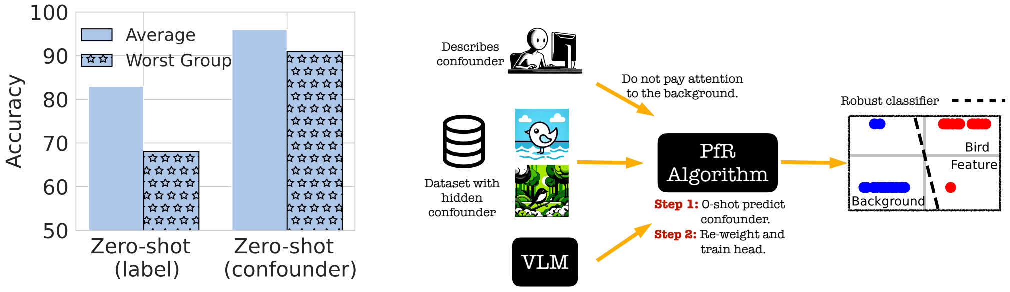
Chapter 12

Prompting is a Double-Edged Sword: Improving Worst-Group Robustness of Foundation Models

Based on Amrith et al. (2024): Amrith Setlur*, Saurabh Garg*, Virginia Smith, Sergey Levine Prompting is a Double-Edged Sword: Improving Worst-Group Robustness of Foundation Models, To appear in Internation Conference on Machine Learning, 2024.

Abstract

In this chapter, we first note that for shifts governed by spurious correlations (features spuriously correlated with the label on the training data, but not on test), the zero-shot and few-shot performance of foundation models is no better than ERM models, and remains unchanged when pretrained data/model size is scaled. Secondly, even in these situations, foundation models are quite accurate at predicting the value of the spurious feature. In a simplified setup, we theoretically analyze both these findings. Specifically, we show that the simplicity bias of foundation models is vulnerable to spurious correlations in contrastive pretraining, and learns features that mostly rely on the spurious attribute, compared to more robust features. We leverage these observations to propose Prompting for Robustness (PfR) which first uses foundation models to zero-shot predict the spurious attribute on labeled examples, and then learns a classifier with balanced performance across different groups of labels and spurious attribute. Across 5 vision and language tasks, we show that PfR's performance nearly equals that of an oracle algorithm (group DRO) that leverages human labeled spurious attributes.



(a) (b)

Figure 12.1: (a): *Foundation models are not robust to spurious correlations, but can predict them*; Averaged across four tasks with spurious correlations, we see that while zero-shot foundation models performance much worse on groups where the spurious correlation is absent, they are highly accurate at predicting the spurious attribute itself, across all groups. (b): *Prompting for Robustness (PfR)*: Leveraging this we propose our method PfR that learns robust classifiers from foundation models in two steps. In Step 1, armed with a text description of the spurious feature, PfR prompts foundation models to zero-shot predict the spurious attribute on a labeled dataset with spurious correlations, and in Step 2 it learns a robust classifier by minimizing worst group loss, across groups given by the combination of the predicted attribute and label.

12.1 Introduction

In this chapter, we show that gains obtained by foundation models in zero-shot prediction on benchmarks like ImageNet with distribution shifts (Radford et al., 2021) do not transfer to other forms of distribution shift such as when confounders that are highly predictive of the label in training distribution are no longer correlated with the label on test (Hall et al., 2023; Tu et al., 2020; Yang et al., 2023). Thus, robustness to hidden confounders in the training data remains an open challenge.

We aim to improve the performance of foundation models on paritions of the distribution (groups) where the confounder is not correlated with the label (minority group). One way is to incorporate downstream labeled data. Unfortunately, unless we have access to deconfounded data (without the spurious correlation), simply fine-tuning naïvely would result in the same issues as standard ERM training, as we confirm experimentally. However, with open-vocabulary foundation models, we can provide for robustness by *telling* the model about the confounder directly (i.e., by describing it in a prompt). One natural way to use this knowledge is to incorporate the description into the classification prompt. However, we observe that even this doesn't improve zero-shot robustness (see Sec. 12.3.2).

We make an intriguing observation: while foundation models are not robust zero-shot classifiers of the true label, they perform remarkably well in predicting the *presence* of spurious attributes. Moreover, we observe that while scaling up the model size and pretraining data does not improve the performance of label prediction on minority groups, the worst group performance of spurious attribute prediction does. Motivated by these findings, we propose a simple technique that we call *Prompting for Robustness (PfR)*. PfR learns robust classifiers for downstream tasks with a few labeled examples and a language description of the confounding attribute. PfR first uses the language description to prompt for a zero-shot classifier that accurately predicts the spurious feature on each labeled example. The value of the label and the predicted confounder jointly define a set of disjoint groups in our data. Then, a robust predictor is learnt by minimizing worst group loss, similar to group DRO, as described by Sagawa et al. (2019), but without ground-truth knowledge of examples in the minority group. This simple method yields surprising performance gains of $\geqslant 40\%$ (averaged across datasets) relative to zero-shot performance of foundation mdoels and ERM on downstream data alone. We further illustrate the applicability of our findings by showcasing its efficacy in extracting group annotations for auditing zero-shot (or ERM) models to assess their robustness. Specifically, we prompt GPT-4V to annotate Chest-Xray 14 dataset (Wang et al., 2017a) for the presence of chest drains (the spurious attribute) and observe a significant robustness gap among ERM models.

Finally, in a simplified setup for multimodal contrastive pretraining, we show that when the spurious correlations in the downstream task are also present in the pretraining distribution over image, and text pairs, then contrastive pretraining learns: (i) image features that couple the spurious feature with other robust features, while placing a higher weight on the spurious one; and (ii) text features that are almost identical for the text descriptions of the label and the spurious attribute. As a consequence of this, we prove that even with infinite pretraining data, the zero-shot performance for the pretrained model would be provably worse than random on examples where label and spurious attributed are uncorrelated. On the other, when it comes to predicting the spurious attribute it has almost perfect accuracy on all examples — precisely the observations we make empirically as well.

In summary our key contributions are as follows. First, we study the performance of foundation models across five vision and language classification tasks with hidden confounders, and observe that while foundation models have poor zero-shot performance on minority examples (that does not improve with scale), they are accurate at predicting the value of the confounder. Second, we leverage this finding to propose a new and simple method: PfR which first zero-shot predicts the confounder when given a text description of it, and then learns a robust classifier across predicted groups. Theoretically, we tie the performance of PfR to the zero-shot accuracy of foundation models on tasks with spurious correlations. Thus, in a simplified setup we provide a theoretical analysis for the zero-shot performance of solutions learned by multimodal contrastive pretraining, and reconcile our theoretical insights with practical findings. Empirically, we show PfR's worst group performance nearly matches the oracle (group DRO) on all datasets.

12.2 Problem setup

We aim to study the robustness of zero/few-shot foundation models, to distribution shifts in classification tasks with spurious correlations. We ground this statement more formally by first defining the task distribution, the model of distribution shift, and what it means to be robust to it.

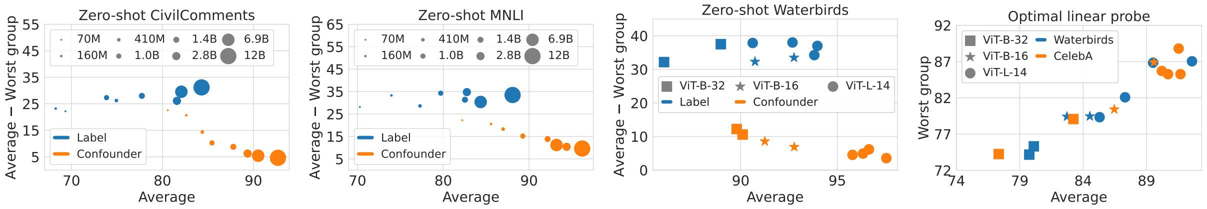


Figure 12.2: *Robustness gap versus average performance as pretraining data and model sizes increase.* We observe that while the robustness gap for confounder prediction decreases the gap between average and worst case increases or remains the same for label prediction.

For a classification task, we use $\mathcal{X}$ to denote input text/image and $\mathcal{Y}$ for the set of labels. Additionally, we also define a set $\mathcal{C}$ of spurious attributes (or confounders). With $\mathscr{G} =: \{G_1, G_2, \ldots, G_k\}$, we define a set of disjoint subsets of $\mathcal{X} \times \mathcal{Y} \times \mathcal{C}$ where each G_i has distribution $P_i(x, y, c)$. Then, our task distribution is a mixture of distributions over $\mathscr{G}$, i.e, $\sum_i \alpha_i P_i(x, y, c)$ where α_i is the proportion of data from each group. In particular, each group G_i corresponds to a unique pair of label and confounder values (y_i, c_i), *i.e.* $\mathbb{1}((x, y, c) \in G_i) = \mathbb{1}(y = y_i)\mathbb{1}(c = c_i)$. When the label y and spurious attribute c are heavily correlated, a classifier that *only* learns the spurious feature c can easily predict the label y. But, this creates a performance disparity across groups where correlations do not hold. For *e.g.* in Waterbirds (Sagawa et al., 2019), the spurious attribute is the background of the bird, the labels are the category of the bird (landbird vs waterbird) and the groups are defined over the joint space of the bird category and its background.

Under distribution P, the average error of a label classifier f is $\text{err}_{\text{y}}^{\text{avg}}(f) =: \mathbb{E}_P\left[\mathbb{1}(f(x) \neq y)\right]$ and spurious atribute classifier g is $\text{err}_{\text{sp}}^{\text{avg}}(g) =: \mathbb{E}_P\left[\mathbb{1}(g(x) \neq c)\right]$. Similarly, their corresponding worst-case error counterparts, taken over groups is: $\text{err}_{\text{y}}^{\text{wg}}(f) =: \max_{G \in \mathscr{G}} \mathbb{E}_{P|G}\left[\mathbb{1}(f(x) \neq y)\right]$ and $\text{err}_{\text{sp}}^{\text{wg}}(g) =: \max_{G \in \mathscr{G}} \mathbb{E}_{P|\mathscr{G}}\left[\mathbb{1}(g(x) \neq c)\right]$. We define the *robustness gap* as the difference between the average case and worst case performance. Consequently, a classifier with low robustness gap for label prediction performs similarly on any distribution that only reweights group proportions α_i. Alternatively, robustness to such group shifts is achieved by having a low robustness gap.

In this work, our goal is to learn a label classifier with (i) high average accuracy, and (ii) low robustness gap. For this, we assume that we are given a text description t_c of the confounder c, along with a few i.i.dlabeled samples $\mathcal{D}$ from $P(x, y)$. Unless specified otherwise, we assume that group annotations are not given to us. Finally, we use FM to denote a foundation model, whose zero-shot prediction of the spurious attribute in input x is $\text{FM}(x, t_c)$.

12.3 Zero-shot robustness of foundation models

In this section, we examine the zero-shot performance of open-vocabulary foundation models on commonly used benchmarks for spurious correlations that involve known confounders.

We find that the zero-shot performance of foundation models suffers from a large robustness gap, indicating a substantial difference between average-case and worst-group performance (as previously demonstrated by Lee et al. (2023); Tu et al. (2020); Yang et al. (2023)). As we increase the scale of pretraining datasets for foundation models, although the models might become better, the robustness gap stays the same or widens, indicating that scale alone does not provide robustness to confounders. Subsequently, we experiment with incorporating a natural language description of the spurious attribute. Our findings indicate that while the inclusion of spurious attribute descriptions through naïve zero-shot prompting does not yield improvements, these models demonstrate high accuracy in predicting the presence of the spurious attribute itself. Building on these findings, we propose our method, Prompting for Robustness (PfR), in the next section.

Finally, we evaluate efficacy of our observation in identifying spurious correlations on a practical task in medical diagnosis. In particular, we annotate Chest Xray-14 dataset for the presence of chest-drain which is known spurious correlation for predicting pneumothorax (Oakden-Rayner et al., 2020). On the annotated groups, ERM models trained on MedCLIP features (Wang et al., 2022a) show large difference between average case and worst case performance. This highlights efficacy of our observation in auditing models to evaluate their robustness to spurious correlations.

12.3.1 Setup

Datasets. We experiment with datasets in both language and vision modalities. For language, we experiment with: (i) MNLI (Williams et al., 2017), where the prediction task is relationship between two input sentences as being contradiction, entailment, or none of the two. Here the spurious attribute is the presence of negation words, e.g., 'no', and 'never'. (ii) CivilComments (Borkan et al., 2019; Koh et al., 2021), where the task is toxicity prediction and the spurious correlation is with the underlying attribute annotating the comment, e.g., male versus female, Christian versus Muslim, etc. For the vision modality, we experiment with: (iii) Waterbirds (Sagawa et al., 2019), where the prediction task is water bird versus land bird classification, and the spurious attribute is the background of the image (i.e., land versus water background); (iv) CelebA (Sagawa et al., 2019), where the prediction task is gender and the spurious attribute is the color of hair. We also experiment with the CXR-drain dataset introduced in Sec. 12.3.3.

Experimental setup. For our zero-shot probing results, we experiment with a number of pretrained foundation models. For vision, we experiment with CLIP (Gadre et al., 2023; Radford et al., 2021). For language, we experiment with RoBerta (Liu et al., 2019b), Llama-2 (Touvron et al., 2023) and Pythia models (Biderman et al., 2023). We also experiment with publicly available models where we vary the model and pretraining dataset sizes in each category. For our ERM experiments, we train linear classifiers on the penultimate layer outputs (representation). For our zero-shot probes, we leverage standard prompts commonly used in the literature. Precise details about prompts used on each dataset are in App. K.3.1.

Evaluation metrics. Along with the prediction accuracy of the label on the worst-case group, we also report average performance. Additionally, we also evaluate the performance of predicting the spurious attribute.

Prompt	Predict	Waterbirds		CelebA		CivilComments		MNLI	
		WG	Avg	WG	Avg	WG	Avg	WG	Avg
Is this label L?	L	59.38	91.97	77.69	81.11	59.25	85.75	76.54	84.79
Is this label L? Ignore confounder C.	L	61.37	92.58	86.73	90.28	52.81	87.41	77.95	80.56
Is this label L and confounder C?	L,C	57.38	88.15	78.54	83.11	54.29	86.60	75.73	82.91
Is this confounder C?	C	90.55	96.33	95.01	99.15	86.73	92.70	92.37	96.19

Table 12.1: *Naively incorporating the confounder description into the label classification prompt does not improve robustness.* Results with leveraging natural language description of the group and label for zero-shot classification.

12.3.2 Observations

Large zero-shot performance gap between the average and worst group. Zero-shot results are in Table 12.2. When evaluating CLIP L/14 models on vision datasets, a notable drop of 32% is observed between average and worst group accuracy on Waterbirds dataset, and a drop of 3.5% is observed on CelebA. Turning to language datasets, the evaluation of the Llama-2 13b model indicates a significant 25% performance decline in CivilComments and a 7% drop in MNLI. Notably, the drops observed here are similar to the performance drops observed with models trained with ERM on their corresponding labeled data (Idrissi et al., 2022; Sagawa et al., 2019). The decline seen with ERM models is typically ascribed to the existence of hidden confounders in the training data (Sagawa et al., 2019), suggesting that pretraining datasets also frequently suffer from analogous spurious correlations. We formalize this intuition in Sec. 12.4.

Incorporating the group description naïvely does not help out of the box. We incorporate spurious attribute description in our zero-shot prompt to predict the label and the spurious attribute jointly. Results are shown in Table 12.1. However, the zero-shot performance for the worst-case group doesn't improve – there is less than a 1% change between the zero-shot and zero-shot with spurious attribute description rows in Table 12.1. We also evaluated other variants, where we explicitly instructed the model to ignore spurious attributes, but this did not substantively impact worst-group performance (details are in App. K.3.2).

Foundation models are surprisingly good at predicting the presence of hidden confounders. Results are in Table 12.1. Instead of incorporating spurious attribute description together with the label, we experiment with predicting the presence of a spurious attribute alone. On all standard spurious correlation benchmarks, we observe that the average performance of predicting the presence of the spurious attribute is around 95% with a similar worst-case group performance. This consistent performance is observed across different groups, emphasizing that, despite foundation models exhibiting significant robustness gaps in the joint prediction of spurious attributes and labels, the predictive accuracy for spurious attributes alone remains superior.

Scaling pretraining datasets and models does not improve zero-shot group robustness. The scaling trend results are presented in Fig. 12.2 (a)-(c), showcasing the performance plotted on average against the difference between average performance and worst-case performance. We analyze this difference in comparison to the average case for both zero-shot label and spurious attribute prediction. As we scale up the pretraining datasets and models, we observe that while the difference reduces for the cofounder prediction, the difference doesn't improve for the label prediction task. This highlights that the prediction performance on standard spurious correlation benchmarks don't improve with scaling and will require post-training interventions.

Scaling pretraining datasets and models does improve underlying representations. As expected we observe that the average and worst-case accuracy (trained with DRO on downstream labeled data) improves as we increase the scale of model size and pretraining data (Fig. 12.2 (d)).

12.3.3 CXR-Drain: Annotating confounders with GPTV-4

In this section, we evaluate the ability to predict spurious correlation in a zero-shot way on a task where ground truth annotations are not publicly available. We choose to annotate 2400 images from Chest Xray-14 dataset (Wang et al., 2017a) for the presence of chest drain with GPT4-V (details are in App. K.3.3). On this dataset, the goal is to predict the whether the patient suffers from pneumothorax disease given their chest x-ray image and the presence of a chest tube in the chest cavity acts as a confounder. It is noteworthy that while previous studies have underscored the issue of spurious correlations in pneumothorax prediction (Oakden-Rayner et al., 2020), the spurious attributes pertinent to this task are not openly available. We refer to the subset of Chest Xray 14 with annotated spurious attributes as CXR-Drain.

While the annotations obtained with GPT4-V are expected to be noisy (different from ground truth annotations for the presence of chest drain), we observe that models trained with ERM show a significant performance gap on the constructed CXR-Drain dataset (Table 12.2). Next, we also note that CXR-drain differs from existing semi-synthetic spurious correlation benchmarks, e.g., the worst group is not the minority group which, and hence, re-weighting based methods (Idrissi et al., 2022; Kirichenko et al., 2022) that simply re-weight different groups may perform poorly when compared with DRO. Due to its unique properties, we believe that CXR-drain will also serve as a crucial benchmark for future research on spurious correlations, and we plan to publicly release the dataset.

12.4 Theoretical analysis of multimodal contrastive pre-training

From Section 12.3, we recall that the worst group zero-shot performance in some cases (like predicting the label of a task with hidden confounders) never improves with scale. So, why does confounder prediction improve? In this section, we analyze both these trends theoretically when pretraining on data where the label is correlated with the confounder, just as the task. We conduct our analysis for multimodal contrastive pretraining. Not only is the contrastive objective more amenable to theoretical analysis, it is commonly used in practice for training some vision-language foundation models (*e.g.* CLIP) that aligns features of image and caption (text) pairs (Radford et al., 2021; Wang et al., 2022a).

Broadly speaking, we show that when certain spurious correlations are also present in pretraining, then contrastive learning only learns image features that heavily couple the spurious feature with other robust features predictive of the label. In this coupling, the component along the spurious feature is higher when the signal-to-noise ratio along the robust feature is poor. Further, the text encoder learns almost identical representations for the confounder and label. As a result, even when trained with infinite pretraining data, we show that the worst group accuracy of the zero-shot label predictor is worse than random, while that of the confounder predictor is nearly perfect.

Setup. The downstream task T has joint distribution $P(x, y, c)$ over image, label and confounder, where both y and c take values in $\{+1, -1\}$ (see (12.1)). Label and confounder are tied by b sampled from a Bernoulli with mean p, where higher p implies stronger correlation between y and c. The input x is split into three components, *i.e.* $x = [x_{\mathrm{r}}, x_{\mathrm{c}}, x_{\mathrm{n}}]$, where $x_{\mathrm{r}} \in \mathbb{R}$ is the robust feature determined solely by label, $x_{\mathrm{c}} \in \mathbb{R}$ by the confounder, $x_{\mathrm{n}} \in \mathbb{R}^{d_{\mathrm{n}}}$ is high dimensional noise.

$$\begin{aligned} y \sim \mathrm{Unif}\{+1, -1\},\ b \sim \mathrm{Bern}(p),\ c = y(2b-1) \\ x_{\mathrm{r}} \sim \mathcal{N}(y, \nabla^2),\ \ x_{\mathrm{c}} = c,\ \ x_{\mathrm{n}} \sim \mathcal{N}(\mathbf{0}_{d_{\mathrm{n}}}, \backslash^2 \mathbf{I}_{d_{\mathrm{n}}}). \end{aligned} \tag{12.1}$$

Contrastive pretraining. The pretraining distribution $Q(x, t)$ for multimodal learning is defined over $\mathcal{X} \times \mathcal{T}$ where $\mathcal{X}$ is the set of images and $\mathcal{T}$ is the set of text inputs. Contrastive pretraining learns an image encoder $\phi : \mathcal{X} \mapsto \mathbb{R}^k$ and a text encoder $\omega : \mathcal{T} \mapsto \mathbb{R}^k$ by pushing together representations of image and text pair sampled from $Q(x, t)$, and pulling apart representations of independent sampled pairs of images from $Q(x)$ and texts from $Q(t)$. We analyze the setting where contrastive pretraining learns ϕ, ω by minimizing spectral contrastive loss (HaoChen et al., 2021):

$$-2\mathbb{E}_{(x,t)\sim Q}\phi(x)^\top \omega(t) + \mathbb{E}_{x \sim Q}\mathbb{E}_{t \sim Q}(\phi(x)^\top \omega(t))^2. \tag{12.2}$$

For simplicity, we consider $Q(x, t)$ that is relevant for the downstream task T. Thus, the set of text descriptions $\mathcal{T}$ is: $\{t_{y,1}, t_{y,-1}, t_{c,1}, t_{c,-1}\}$. The marginal $Q(t)$ is uniform. For the conditionals, given $a \in \{-1, 1\}$, $Q(x \mid t_{y,a}) = P(x \mid y = a)$, and $Q(x \mid t_{c,a}) = P(x \mid c = a)$. Note that, as p in (12.1) increases, not only does it increase downstream correlation $\mathbb{E}_P[yc]$, it also increases the overlap between $Q(x \mid t_{y,a})$ and $Q(x \mid t_{c,a})$ in the pretraining distribution.

Zero-shot predictors. In practice, pretrained ϕ, ω are used as zero-shot classifiers by evaluating $\phi(x)^\top \omega(t)$, where t is the labels's text description. Adhering to this, we define zero-shot label classifier $f =: 2 \cdot \mathbb{1}(\phi(x)^\top(\omega(t_{y,1}) - \omega(t_{y,-1})) \geqslant 0) - 1$, and zero-shot confounder classifier $g =: 2 \cdot \mathbb{1}(\phi(x)^\top(\omega(t_{c,1}) - \omega(t_{c,-1})) \geqslant 0) - 1$.

12.4.1 Key insights and main result.

In Theorem 12.4.1 we provide an informal statement of our main result on the worst group zero-shot performance of label and confounder classifiers. We note that as the spurious correlation p increases, the worst group error worsens for the label predictor and on the other end, improves for the confounder predictor.

Theorem 12.4.1. *(zero-shot robustness; informal) Let the zero-shot label (f) and confounder classifier (g) be obtained by minimizing the loss in (12.2) on infinite pretraining data for linear functions ϕ, ω. Then, for $\nabla = \Omega(1)$, label classifier is worse than random on the worst group, since $\mathrm{err}_{\mathrm{y}}^{\mathrm{wg}}(f) = {}^{1}\!/_{2}\,\mathrm{erfc}(-c_1 p \nabla)$. On the other hand, the confounder classifier suffers small error on all groups since $\mathrm{err}_{\mathrm{sp}}^{\mathrm{wg}}(g) = {}^{1}\!/_{2}\,\mathrm{erfc}(c_2 p \nabla)$. Here, $c_1, c_2 > 0$ are constants.*

Our analysis in 12.4.2 will show that the above result is a consequence of (i) image encoder relying more on non-robust compared to robust x_{r} when ∇ is higher; (ii) text encoder failing to learn separate representations for the label and confounder descriptions.

Intuition. During multimodal contrastive pretraining feature alignment of the image and corresponding text features is achieved when images $x_i, x_j \sim Q(x \mid t)$ sampled from the text have well clustered representations, and the clusters of different text inputs are well separated. Our understanding relies on two key observations. First, when the pretraining distribution replicates the task distribution's spurious correlations (as $Q(x,t)$ does with $P(x,y,c)$), then the clusters learned for the label and confounder necessarily overlap since $Q(x \mid t_{y,a}) \approx Q(x \mid t_{c,a})$ (matches on all but the group where correlation is absent). Thus, given this distribution overlap the optimal text encoder's features for the label and the confounder would be very similar. Second, when the noise along the robust feature ∇ is high, the intra cluster variance along the non-robust feature x_{c} is relatively lower. This biases contrastive learning to place higher weight on the non-robust feature, in learning features that separate clusters corresponding to the different text inputs with large margins. Together, this would lead to poor robustness for the label predictor, and opposite for the spurious attribute predictor, as we note in Theorem 12.4.1.

12.4.2 Optimal solutions for spectral contrastive loss.

In this subsection, we present Theorem 12.4.2 which states the solutions for the image and text encoders learned by minimizing the objective in (12.2), for linear ϕ and $k = 2$. In Appendix K.2.2 we prove results for more general families. We make two observations that are consistent with our intuition above. First, we see that when the noise along robust feature (∇) is large, then any increase in spurious correlation (p), increases the optimal image features' weights along spurious atttribute (x_{c}), as θ decreases. Second, we see that the optimal solution for the text learns identical features for label and confounder. Thus, on any group that they disagree, the upweighted x_{c} feature contributes more to the prediction.

Theorem 12.4.2 (Optimal solutions for (12.2); informal)**.** *Let $\phi(x) = [\phi_1^\top x, \phi_2^\top x]$ for $\phi_1, \phi_2 \in \mathbb{R}^d$. When $p > 0.5, \nabla = \Omega(1)$, the optimal values for norm bounded ϕ_1, ϕ_2 that* *minimize the objective in* (12.2)*, are*

$$\phi_1 = \left[{}^{\cos(\theta)}/\sqrt{\nabla^2+1}, \sin(\theta)\right]^\top, \quad \textit{and}$$

$$\phi_2 = \left[-{}^{\sin(\theta)}/\sqrt{\nabla^2+1}, \cos(\theta)\right]^\top,$$

where $\theta = {}^1/{}_{p\nabla^2}$*. Also, the text features match for label and confounder,* i.e. $\omega(t_{y,a}) = \omega(t_{c,a}) = [1, a]^\top$ *for* $a \in \{1, -1\}$*.*

12.5 Prompting for Robustness

Our results in Section 12.3 suggest that zero-shot classification with foundation models often attains high average group accuracy but low worst-group accuracy. However, we note that they are surprisingly accurate at predicting the presence of a confounder. We leverage this finding to propose a simple but effective method: Prompting for Robustness (PfR). PfR learns a robust classifier given a few labeled examples and a text description of the confounder. While standard techniques of using labeled data or foundation model alone fail, we show that PfR efficiently uses both to recover a classifier with worst group performance close to that of methods that have ground truth group information (i.e., Group DRO).

Method	**Waterbirds**		**CelebA**		**CivilComments**		**MNLI**		**CXR-Drain**	
	WG	Avg	WG	Avg	WG	Avg	WG	Avg	WG	Avg
Zero-shot	59.38	91.97	77.69	81.11	59.25	85.75	76.54	84.79	–	–
ERM	70.71	98.75	54.84	94.96	61.35	92.42	67.30	87.71	51.79	76.10
JTT	85.86	95.47	82.49	92.74	72.73	90.54	72.75	86.73	56.52	77.53
Yang et al.	90.13	95.80	**88.12**	91.64	–	–	–	–	59.37	74.58
Zhang et al.	86.90	96.20	84.60	90.40	50.10	54.20	–	–	–	–
PfR (ours)	**91.05**	94.32	88.05	91.97	**77.83**	88.70	**81.28**	84.60	**68.55**	76.73
Group DRO (oracle)	93.23	94.40	90.79	92.32	80.21	86.52	81.54	84.37	–	–

Table 12.2: *PfR improves worst group performance over ERM and zero-shot foundation models:* On five benchmarks from Section 12.3 we evaluate average and worst-group performance of PfR and compare it with baselines JTT, ERM, and zero-shot.

Prompting for Robustness (PfR). PfR (summarized in Algorithm 10) runs in two stages. In the first stage, PfR prompts an open vocabulary foundation model FM with the text description t_c of the confounding attribute and recovers a zero-shot prediction of the confounder c on any given input (for *e.g.* in the case of CivilComments the confounder is described as "race, religion or gender"). Using this, each training example (x_i), which was previously annotated only for the label of interest (y_i), is additionally annotated with the value of the confounding attribute $(\widehat{c}_i)$ (for *e.g.* "black/white and christian/muslim"). The training dataset is then split into disjoint groups $\widehat{\mathscr{G}}$ based on the paired value $(y_i, \widehat{c}_i)$ of the label and predicted confounder. In the second stage, PfR learns a robust classifier by minimizing the worst group loss over each predicted group, minimizing

$$\min_f \max_{G \in \widehat{\mathscr{G}}} \; \mathbb{E}\left[\ell(f(x), y) \mid x \in G\right]. \tag{12.3}$$

Algorithm 10 Prompting for Robustness (PfR)

```
**Input:** Foundation model FM, text description of counfounder $t_c$, labeled dataset
$\mathcal{D} = \{x_i, y_i\}_{i=1}^n$.
Stage I: Predict confounder (spurious attribute)
• Prompt FM with $t_c$ to get zero-shot head $\mathrm{FM}(\cdot, t_c)$.
• For each datapoint predict confounder $\widehat{c}_i \leftarrow \widehat{\mathrm{FM}}(x_i, t_c)$.
• Partition dataset into set of disjoint groups $\widehat{\mathscr{G}}$ based on value of label and predicted
  confounder: $(y, \widehat{c})$.
Stage II: Optimize worst group loss with DRO
• Learn robust classifier $f$ by minimizing the worst loss over predicted groups in (12.3).
```

The above objective can be optimized with an online algorithm that treats f and G as players in a minimax game, analogously to the group DRO algorithm described by Sagawa et al. (2020). Hence, we reuse their Algorithm 1 to optimize our objective in Equation (12.3). The key difference between our objective and standard Group DRO is that the latter minimizes worst group loss over ground truth groups obtained by using human annotations of the confounder attribute. Based on our findings from Section 12.3, we should expect that the confounder can be predicted accurately in zero shot, enabling PfR to possibly match the performance of Group DRO. This is indeed what we will see in experiments.

12.5.1 PfR is more robust than zero-shot and ERM

On the five datasets we introduced previously, we evaluate the performance of PfR and compare with both zero-shot and few-shot algorithms that have access to a few labels (but not the ground-truth group labels).

Setup and baselines. On the language tasks we use Llama2-7b and Llama2-13b models (Touvron et al., 2023) for zero-shot prediction (reporting max of the two), and on the vision tasks we use CLIP-ViT-L/16 (Radford et al., 2021). We compare to JTT (Liu et al., 2021a), a prior method for robustness that does not require group labels, as well as standard ERM. We also include Group DRO (Sagawa et al., 2019) as an oracle baseline that has access to true group labels. All few-shot methods including PfR are used to train a linear head over fixed features. In the language task we train a linear head on top of features learned by finetuning a RoBERTa encoder (Liu et al., 2019b) on the MNLI/CivilComments dataset, and for vision tasks we train a linear head over CLIP's image encoder.

Results. In Table 12.2, we compare average and worst group performance for different methods. First, we observe that averaged across datasets, PfR reduced worst group error by 47% compared to zero-shot, and 52% and 30% compared to ERM and JTT, respectively. On some datasets like Waterbirds, the worst group gains are as high as > 75%. More importantly, PfR's performance closely matches that of the oracle Group DRO algorithm across all datasets. Additionally, unlike overly pessimistic DRO objectives like CVaR-DRO (Hu et al., 2018), the average performance is not significantly compromised from trying to improve worst group accuracy. Thus, we see that PfR learns a classifier robust to spurious correlations without much human annotation overhead beyond a description of

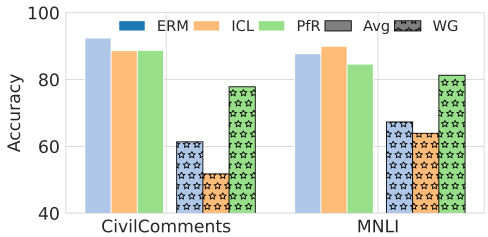


Figure 12.3: *In-context learning with 128 examples does not improve robustness gap, instead hurts it:* Average and worst-group performance of ICL, ERM and PfR on language tasks.

the confounder.

12.5.2 Comparing PfR with in-context learning

For language tasks, in-context learning (ICL) is a commonly used few-shot method to improve performance when zero-shot methods are poor (Brown et al., 2020). In ICL, some labeled training examples are fed along with a language description of the classification task to large language models (*e.g.* GPT-3.5, Llama). Since PfR also uses labeled examples, we compare our method with ICL on CivilComments and MNLI (see Fig. 12.3). We observe that while ICL improves over zero-shot inference on average, the worst-group performance remains almost unchanged for CivilComments and worsens for MNLI. We can therefore see that ICL is not a viable alternative to PfR. One reason for why ICL can hurt worst group performance is prior works have shown ICL in language models to make predictions consistent with ERM models trained with gradient descent (Ahn et al., 2023; Akyürek et al., 2022; Von Oswald et al., 2023). Since such ERM models are known to latch onto spurious correlations in the training data (Nagarajan et al., 2020; Shah et al., 2020), we would expect ICL to improve average performance at the expense of worst group performance.

12.5.3 Theoretical analysis of PfR

PfR relies on foundation models to accurate predict the confounding attribute (Sec. 12.3), even when they cannot in zero shot disentangle this confounder from the class label. Given the description t_c, the confounder prediction error suffered by the zero-shot model in the first stage of PfR is $\mathrm{err}_c(\mathrm{FM}(\cdot, t_c))$. In Theorem 12.5.1 we provide worst-group generalization error guarantees for PfR.

Theorem 12.5.1 (PfR's worst group error; informal)**.** *For PfR output $\widehat{f}$, w.h.p. $1-\delta$, worst group generalization error of $\widehat{f}$ is $\lesssim \sqrt{{}^{\log \mathfrak{C}(\mathcal{F})K/\delta}/_{n}} + \mathrm{err}_c(\mathrm{FM}(t_c))$, where $\mathfrak{C}(\mathcal{F})$ is complexity of $\mathcal{F}$, K is number of groups and latter term is* FM*'s zero-shot performance on confounder prediction.*

The above result shows that the worst group accuracy of PfR is upper bounded by two terms. The first term is the generalization error suffered by the oracle algorithm (Group DRO), and the second is the zero-shot error in predicting the confounder. Thus, as the the zero-shot accuracy of confounder prediction improves, it linearly affects worst-group error guarantees for PfR.

12.6 Related Work

Several prior works use distribution robust optimization (DRO) to learn predictors robust to shifts in an uncertainty set (Ben-Tal et al., 2013; Blanchet and Murthy, 2019; Duchi et al., 2016; Duchi and Namkoong, 2021). For spurious correlation problems that result in more specific group shifts, DRO tends to be overly pessimistic (worse than ERM) (Hu et al., 2018). To address this, previous works assume knowledge of the spurious attribute, and either only minimize worst loss over known groups (Sagawa et al., 2019) or average loss over re-weighted ones (Idrissi et al., 2022; Kirichenko et al., 2022). Since it is restrictive to assume group knowledge, other works used relied on two observations: spurious attributes are easier to learn (than robust features) and ERM suffers from a simplicity bias (Sagawa et al., 2020; Shah et al., 2020). Using this, they either reconfigure DRO's uncertainty set (Setlur et al., 2023) (or make it random (Zhai et al., 2021)), while other works (Liu et al., 2021a; Nam et al., 2020) exploit it to recover the hidden minority group with ERM losses. Finally, some other works on robustness to hidden confounders (Bao and Barzilay, 2022; Creager et al., 2021; Sohoni et al., 2021) either rely on dataset dependent heuristics, or the ability to query test samples (Lee et al., 2022). Different from the above, we assume a language description of the confounder (as opposed to groups). Armed with this, we use open vocabulary models to predict the presence of a confounder, and then learn robust predictors with DRO over predicted groups. Thus, while we leverage DRO formulation for robustness guarantees, we also avoid its pitfalls by relying on zero-shot foundation models.

12.7 Conclusion and Limitations

In this work, we focus on the robustness of zero-shot models to tasks with spurious correlations. While foundation models have shown unprecedented zero-shot capabilities, we show that these models struggle when confounders lose correlation with labels. To address this, we propose Prompting for Robustness (PfR), leveraging language descriptions to prompt zero-shot classifiers and train robust models. Empirical results reveal significant performance gains in the worst accuracy groups. Overall, this work offers insights and a practical approach to enhance foundation model robustness against hidden confounders, contributing to bias mitigation and improved fairness in machine learning.

Chapter 13

Conclusion and Future Work

In conclusion, the work in this thesis demonstrates the pivotal role of leveraging unlabeled data to enhance the robustness and adaptability of deep learning models, addressing critical challenges posed by distribution shifts in real-world applications. A common theme across my work involves leveraging unlabeled data to aid in uncertainty estimation and self-training to improve classification models under distribution shift. Uncertainty estimation plays a crucial role in decision-making, while self-training enables models to improve themselves iteratively. These techniques have shown promise in various distribution shift scenarios, showcasing their potential to enhance model performance and robustness. As we are building strong foundation models, the insights gained here pave a way for exciting future work in the following directions:

- **Understanding and improving uncertainty estimation in LLMs and VLMs:** Uncertainty estimation for natural language tasks with generative foundation models presents interesting challenges and opportunities for future research. Question answering tasks or abstractive tasks, such as text summarization or paraphrasing, often require models to generate novel and coherent outputs, making uncertainty estimation crucial for assessing the reliability of these outputs. LLMs demonstrate the capability to predict uncertainty by emitting confidence scores in predictions, which are often calibrated (Kadavath et al., 2022; Tian et al., 2023). However, the underlying factors influencing this capability are not fully understood. Moreover, it remains unclear how these capabilities are transferred to VLMs obtained by instruction fine-tuning of LLMs. Future research could delve deeper into elucidating these factors, including the impact of input data characteristics and training dynamics. By gaining insights into the limitations and biases inherent in current uncertainty estimation techniques, efforts can be directed towards enhancing the interpretability and calibration of uncertainty scores. This understanding can pave the way for improved techniques for uncertainty estimation in LLMs. By accurately quantifying uncertainty associated with generated abstractions, these advancements could lead to more reliable and contextually appropriate outputs from generative foundation models.

- **Self-training to improve reasoning capabilities in LLMs:** Self-training involves leveraging unlabeled data to iteratively improve model performance, which could be particularly impactful for fine-tuning reasoning abilities in these models. One avenue of investigation could involve designing self-training strategies that focus on refining reasoning skills through exposure to diverse and contextually rich unlabeled text data (Chen et al., 2024). By iteratively updating the model based on self-generated labels from the most confident predictions, coupled with human verification and feedback loops, it may be possible to boost the model's ability to infer logical relationships, draw nuanced conclusions, and generalize effectively across different tasks and domains. This approach has the potential to improve reasoning abilities in LLMs beyond what is achievable by pretraining alone.

- **Continual Training of LLMs:** Building on our work on continual training with CLIP (Garg et al., 2024), a interesting veneue to expand our investigation is generative language and vision models. Large language models (LLMs), in particular, are prone to "hallucination" of factually incorrect information on queries involving dynamically evolving concepts, e.g., GPT-3.5's and Llama-2-70b-chat response to "Who is 56th prime minister of United Kingdom?" is Boris Johnson (which is incorrect as Rishi Sunak is 56th prime minister). How can we make these models more robust to the evolving nature of the world?

  While one obvious solution is to keep these models up to date on the latest data via retraining, this solution is extremely compute inefficient. An alternate approach would be to instead develop approaches that allow us to quickly adapt the parameters of a foundation model based on limited amounts of new data, without requiring a fresh training run. Recently, researchers have started exploring solutions to the adaptation of foundation models. For LLMs, Vu et al. (2023) introduced FreshLLMs highlighting the need to update LLMs to evolving factual information and proposed a simple method that involves augmenting the input query with corresponding results retrieved from an internet search. This solution doesn't involve any learning/updating LLMs and hence even for repeated queries, such a system can substantially increase the inference time and cost of the deployed system. Another interesting question is around investigating learning rate schedules that are more amenable to continual learning while training models that are deployable at intermediate time steps.

Part V

Appendix

Appendix A

Appendix: A Unified View of Label Shift Estimation

A.1 MLLS Algorithm

Algorithm 11 Maximum Likelihood Label Shift estimation

```
input : Labeled validation samples from source and unlabeled test samples from target.
    Trained blackbox model $\widehat{f}$, model class $\mathcal{G}$ and loss function $l$ for calibration (for instance,
    MSE or negative log-likelihood).
1: On validation data minimize the loss $l$ over class $\mathcal{G}$ to obtain $f = g \circ \widehat{f}$.
2: Solve the optimization problem (2.5) using $f$ to get $\widehat{w}$.
output : MLLS estimate $\widehat{w}$
```

Step 1. description. Let the model class used for *post-hoc calibration* be represented by $\mathcal{G}$. Given a validation dataset $\{(x_{v1}, y_{v1}), \ldots, (x_{vn}, y_{vn})\}$ sampled from the source distribution P_s we compute, $\{(\widehat{f}(x_{v1}), y_{v1}), (\widehat{f}(x_{v2}), y_{v2}), \ldots, (\widehat{f}(x_{vn}), y_{vn})\}$, applying our classifier $\widehat{f}$ to the data. Using this we estimate a function,

$$\widehat{g} = \arg\min_{g \in \mathcal{G}} \sum_{i=1}^{n} \ell(g \circ \widehat{f}(x_{vi}), y_{vi}) \,, \tag{A.1}$$

where the loss function ℓ can be the negative log-likelihood or squared error. Experimentally we observe same performance with both the loss functions. Subsequently, we can apply the calibrated predictor $\widehat{g} \circ \widehat{f}$.

Our experiments follow Alexandari et al. (2021), who leverage BCTS [1] to calibrate their models. BCTS extends temperature scaling (Guo et al., 2017) by incorporating per-class

[1] Motivated by the strong empirical results in Alexandari et al. (2021), we use BCTS in our experiments as a surrogate for canonical calibration.

bias terms. Formally, a function $g : \Delta^{k-1} \mapsto \Delta^{k-1}$ in the BCTS class $\mathcal{G}$, is given by

$$g_j(x) = \frac{\exp\left[\log(x_j)/T + b_j\right]}{\sum_i \exp\left[\log(x_i)/T + b_i\right]} \quad \forall j \in \mathcal{Y}$$

where $\{T, b_1, \ldots, b_{|\mathcal{Y}|}\}$ are the $|\mathcal{Y}| + 1$ parameters to be learned.

A.2 Prior Work on Label Shift Estimation

Dataset shifts are predominantly studied under two scenarios: covariate shift and label shift (Storkey, 2009). Schölkopf et al. (2012) articulates connections between label shift and covariate shift with anti-causal and causal models respectively. Covariate shift is well explored in past (Cortes and Mohri, 2014; Cortes et al., 2010; Gretton et al., 2009; Zadrozny, 2004; Zhang et al., 2013).

Approaches for estimating label shift (or prior shift) can be categorized into three classes:

1. Methods that leverage Mixture Proportion Estimation (MPE) (Blanchard et al., 2010; Ramaswamy et al., 2016) techniques to estimate the target label distribution. MPE estimate in general (e.g. Blanchard et al. (2010)) needs explicit calculations of $p_s(x|y)(= p_t(x|y))$ which is infeasible for high dimensional data. More recent methods for MPE estimation, i.e. Ramaswamy et al. (2016), uses Kernel embeddings, which like many kernel methods, require the inversion of an $n \times n$ Gram matrix. The $\mathcal{O}(n^3)$ complexity makes them infeasible for large datasets, practically used in deep learning these days;

2. Methods that directly operate in RKHS for distribution matching (Du Plessis and Sugiyama, 2014b; Zhang et al., 2013). Zhang et al. (2013) extend the kernel mean matching approach due to Gretton et al. (2009) to the label shift problem. Instead of minimizing maximum mean discrepancy, Du Plessis and Sugiyama (2014b) explored minimizing PE divergence between the kernel embeddings to estimate the target label distribution. Again, both the methods involve inversion of an $n \times n$ kernel matrix, rendering them infeasible for large datasets; and

3. Methods that work in low dimensional setting (Azizzadenesheli et al., 2019; Lipton et al., 2018b; Saerens et al., 2002) by directly estimating $p_t(y)/p_s(y)$ to avoid the curse of dimensionality. These methods leverage an off-the-shelf predictor to estimate the label shift ratio.

In this paper, we primarily focus on unifying methods that fall into the third category.

A.3 Marginal calibration is insufficient to achieve consistency

In this section, we will illustrate insufficiency of *marginal calibration* to achieve consistency. For completeness, we first define margin calibration:

Definition A.3.1 (Marginal calibration)**.** *A prediction model $f : \mathcal{X} \mapsto \Delta^{k-1}$ is marginally calibrated on the source domain if for all $x \in \mathcal{X}$ and $j \in \mathcal{Y}$,*

$$\mathrm{P}_s(y = j|f_j(x)) = f_j(x)\,.$$

Intuitively, this definition captures per-label calibration of the classifier which is strictly less restrictive than requiring canonical calibration. In the example, we construct a classifier on discrete $\mathcal{X}$ which is marginally calibrated, but not canonically calibrated. With the constructed example, we show that the population objective (2.4) yields inconsistent estimates.

Example. Assume $\mathcal{X} = \{x_1, x_2, x_3, x_4, x_5, x_6\}$ and $\mathcal{Y} = \{1, 2, 3\}$. Suppose the predictor $f(x)$ and $P_s(y|f(x))$ are given as,

$f(x)$	y=1	y=2	y=3
x_1	0.1	0.2	0.7
x_2	0.1	0.7	0.2
x_3	0.2	0.1	0.7
x_4	0.2	0.7	0.1
x_5	0.7	0.1	0.2
x_6	0.7	0.2	0.1

$P_s(y\|f(x))$	y=1	y=2	y=3
x_1	0.2	0.1	0.7
x_2	0.0	0.8	0.2
x_3	0.1	0.2	0.7
x_4	0.3	0.6	0.1
x_5	0.8	0.0	0.2
x_6	0.6	0.3	0.1

Clearly, the prediction $f(x)$ is marginally calibrated. We have one more degree to freedom to choose, which is the source marginal distribution on $\mathcal{X}$. For simplicity let's assume $p_s(x_i) = 1/6$ for all $i = \{1, \ldots, 6\}$. Thus, we have $p_s(y = j) = 1/3$ for all $j = \{1, 2, 3\}$. Note, with our assumption of the source marginal on x, we get $P_t(x_i|y = j) = P_s(x_i|y = j) = P_s(y = j|f(x_i))/2$. This follows as $x \mapsto f(x)$ is an one-to-one mapping.

Now, assume a shift i.e. prior on $\mathcal{Y}$ for the target distribution of the form $[\alpha, \beta, 1 - \alpha - \beta]$. With the label shift assumption, we get

$$\forall i \qquad p_t(x_i) = \frac{1}{2}\left(\alpha P_s(y = 1|f(x_i)) + \beta P_s(y = 2|f(x_i)) + (1 - \beta - \alpha) P_s(y = 3|f(x_i))\right)\,.$$

Assume the importance weight vector as w. Clearly, we have $w_1 + w_2 + w_3 = 3$. Re-writing the population MLLS objective (2.4), we get the maximisation problem as

$$\arg\max_w \sum_{i=1}^{6} p_t(x_i) \log(f(x_i)^T w)\,. \tag{A.2}$$

Differentiating (A.2) with respect to w_1 and w_2, we get two high order equations, solving which give us the MLLS estimate w_f. To show inconsistency, it is enough to consider one instantiation of α and β such that $|3\alpha - w_1| + |3\beta - w_2| + |w_1 + w_2 - 3\alpha - 3\beta| \neq 0$. Assuming $\alpha = 0.8$ and $\beta = 0.1$ and solving (A.2) using numerical methods, we get $w_f = [2.505893, 0.240644, 0.253463]$. As $w = [2.4, 0.3, 0.3]$, we have $w_f \neq w$ concluding the proof.

A.4 Proofs from Section 2.4

Lemma A.4.1 (Identifiability)**.** *If the set of distributions $\{p(z|y) : y = 1, ..., k\}$ are linearly independent, then for any w that satisfies (2.2), we must have $w = w^*$. This condition is also necessary in general: if the linear independence does not hold then there exists a problem instance where we have $w, w^* \in \mathcal{W}$ satisfying (2.2) while $w \neq w^*$.*

Proof. First we prove sufficiency. If there exists $w \neq w^*$ such that (2.2) holds, then we have $\sum_{y=1}^{k} p_s(z, y)(w_y - w_y^*) = 0$ for all $z \in \mathcal{Z}$. As $w - w^*$ is not the zero vector, $\{p_s(z, y), y = 1, ..., k\}$ are linearly dependent. Since $p_s(z, y) = p_s(y)p(z|y)$ and $p_s(y) > 0$ for all y (by assumption), we also have that $\{p(z|y), y = 1, ..., k\}$ are linearly dependent. By contradiction, we show that the linear independence is necessary.

To show necessity, assume $w_y^* = \frac{1}{kp_s(y)}$ for $y = 1, ..., k$. We know that w^* satisfies (2.2) by definition. If linear independence does not hold, then there exists a vector $v \in \mathbb{R}^k$ such that $v \neq 0$ and $\sum_{y=1}^{k} p_s(z, y)v_y = 0$ for all $z \in \mathcal{Z}$. Since the w^* we construct is not on the boundary of $\mathcal{W}$, we can scale v such that $w^* + \alpha v \in \mathcal{W}$ where $\alpha \geqslant 0$ and $v \neq 0$. Therefore, setting $w = w^* + \alpha v$ gives another solution for (2.2), which concludes the proof. □

If f is calibrated, then the two objectives (2.3) and (2.4) are identical when $\mathcal{Z}$ is chosen as Δ^{k-1} and $p(z|x)$ is defined to be $\delta_{f(x)}$.

Proof. The proof follows a sequence of straightforward manipulations. In more detail,

$$\begin{aligned}
\mathbb{E}_t\left[\log f(x)^T w\right] &= \int p_t(x) \log[f(x)^T w] dx \\
&= \int\int p_t(x)p(z|x) \log[f(x)^T w] dx dz \\
&= \int\int p_t(x)p(z|x)\mathbb{I}f(x) = z \log[f(x)^T w] dx dz \\
&= \int\int p_t(x)p(z|x) \log[z^T w] dx dz \\
&= \int p_t(z) \log[z^T w] dz \\
&= \int p_t(z) \log\left[\sum_{y=1}^{k} p_s(y|z) w\right] dz \,,
\end{aligned}$$

where the final step uses the fact that f is calibrated.

□

Theorem 2.4.2 (Population consistency of MLLS)**.** *If a predictor $f : \mathcal{X} \mapsto \Delta^{k-1}$ is calibrated and the distributions $\{p(f(x)|y) : y = 1, \ldots, k\}$ are strictly linearly independent, then w^* is the unique maximizer of the MLLS objective (2.4).*

Proof. According to Lemma 2.4.3 we know that maximizing (2.4) is the same as maximizing (2.3) with $p(z|x) = \delta_{f(x)}$, thus also the same as minimizing the KL divergence between $p_t(z)$ and $p_w(z)$. Since $p_t(z) \equiv p_{w^*}(z)$ we know that w^* is a minimizer of the KL divergence such that the KL divergence is 0. We also have that $\mathrm{KL}(p_t(z), p_w(z)) = 0$ if and only if $p_t(z) \equiv p_w(z)$, so all maximizers of (2.4) should satisfy (2.2). According to Lemma 2.4.1, if the strict linear independence holds, then w^* is the unique solution of (2.2). Thus w^* is the unique maximizer of (2.4).

□

Proposition A.4.2. *For a calibrated predictor f, the following statements are equivalent:*

(1) $\{p(f(x)|y) : y = 1, \ldots, k\}$ are strictly linearly independent.

(2) $\mathbb{E}_s\left[f(x)f(x)^T\right]$ is invertible.

(3) The soft confusion matrix of f is invertible.

Proof. We first show the equivalence of (1) and (2). If f is calibrated, we have $p_s(f(x))f_y(x) = p_s(y)p(f(x)|y)$ for any x, y. Then for any vector $v \in \mathbb{R}^k$ we have

$$\sum_{y=1}^{k} v_y p(f(x)|y) = \sum_{y=1}^{k} \frac{v_y}{p_s(y)} p_s(y)p(f(x)|y) = \sum_{y=1}^{k} \frac{v_y}{p_s(y)} p_s(f(x))f_y(x) = p_s(f(x)) \sum_{y=1}^{k} \frac{v_y}{p_s(y)} f_y(x)\,. \tag{A.3}$$

On the other hand, we can have

$$\mathbb{E}_s\left[f(x)f(x)^T\right] = \int f(x)f(x)^T p_s(f(x))d(f(x))\,. \tag{A.4}$$

If $\{p(f(x)|y) : y = 1, \ldots, k\}$ are linearly dependent, then there exist $v \neq 0$ such that (A.3) is zero for any x. Consequently, there exists a non-zero vector u with $u_y = v_y/p_s(y)$ such that $u^T f(x) = 0$ for any x satisfying $p_s(f(x)) > 0$, which means $u^T\mathbb{E}_s\left[f(x)f(x)^T\right]u = 0$ and thus $\mathbb{E}_s\left[f(x)f(x)^T\right]$ is not invertible. On the other hand, if $\mathbb{E}_s\left[f(x)f(x)^T\right]$ is non-invertible, then there exist some $u \neq 0$ such that $u^T\mathbb{E}_s\left[f(x)f(x)^T\right]u = 0$. Further as $u^T\mathbb{E}_s\left[f(x)f(x)^T\right]u = \int u^T f(x)f(x)^T u \; p_s(x)dx = \int \left|f(x)^T u\right| p_s(x)dx$. As a result, the vector v with $v_y = p_s(y)u_y$ satisfies that (A.3) is zero for any x, which means $\{p(f(x)|y) : y = 1, \ldots, k\}$ are not strictly linearly independent.

Let C be the soft confusion matrix of f, then

$$\begin{aligned} C_{ij} = p_s(\widehat{y} = i, y = j) &= \int d(f(x))\, f_i(x)p(f(x)|y = j)p_s(y = j) \\ &= \int f_i(x)f_j(x)p_s(f(x))d(f(x))\,. \end{aligned}$$

Therefore, we have $C = \mathbb{E}_s\left[f(x)f(x)^T\right]$, which means (2) and (3) are equivalent.

□

We introduce some notation before proving consistency. Let $\mathscr{P} = \{\langle f, w\rangle | w \in \mathcal{W}\}$ be the class of densities[2] for a given calibrated predictor f. Suppose $\widehat{p}_n, p_0 \in \mathscr{P}$ are densities corresponding to MLE estimate and true weights, respectively. We use $h(p_1, p_2)$ to denote the Hellinger distance and $\mathrm{TV}(p_1, p_2)$ to denote the total variation distance between two densities p_1, p_2. $H_r(\delta, \mathscr{P}, P)$ denotes δ-entropy for class $\mathscr{P}$ with respect to metric $L_r(P)$. Similarly, $H_{r,B}(\delta, \mathscr{P}, P)$ denotes the corresponding bracketing entropy. Moreover, P_n denotes the empirical random distribution that puts uniform mass on observed samples $x_1, x_2, \ldots x_n$. Before proving consistency we need to re-state two results:

Lemma A.4.3 (Lemma 2.1 (van de Geer, 2000))**.** *If P is a probability measure, for all $1 \leqslant r < \infty$, we have*

$$H_{r,B}(\delta, \mathscr{G}, P) \leqslant H_\infty(\delta/2, \mathscr{G}) \qquad \text{for all } \delta > 0\,.$$

Lemma A.4.4 (Corollary 2.7.10 (van der Vaart and Wellner, 1996))**.** *Let $\mathcal{F}$ be the class of convex functions $f : C \mapsto [0,1]$ defined on a compact, convex set $C \subset \mathbb{R}^d$ such that $|f(x) - f(y)| \leqslant L \|x - y\|$ for every x,y. Then*

$$H_\infty(\delta, \mathcal{F}) \leqslant K \left(\frac{L}{\delta}\right)^{d/2},$$

for a constant K that depends on the dimension d and C.

We can now present our proof of consistency, which is based on Theorem 4.6 from van de Geer (2000):

Lemma A.4.5 (Theorem 4.6 (van de Geer, 2000))**.** *Let $\mathscr{P}$ be convex and define class $\mathscr{G} = \left\{\frac{2p}{p+p_0} | p \in \mathscr{P}\right\}$. If*

$$\frac{1}{n} H_1(\delta, \mathscr{G}, P_n) \to_P 0\,, \tag{A.5}$$

then $h(\widehat{p}_n, p_0) \to 0$ almost surely.

Theorem 2.4.3 (Consistency of MLLS)**.** *If f satisfies the conditions in Theorem H.2.1, then $\widehat{w}_f$ in (2.5) converges to w^* almost surely.*

Proof. Assume the maximizer of (2.5) is $\widehat{w}_f$ and $p_0 = \langle f, w^*\rangle$. Define class $\mathscr{G} = \left\{\frac{2p}{p+p_0} | p \in \mathscr{P}\right\}$. To prove consistency, we first bound the bracketing entropy for class $\mathscr{G}$ using Lemma A.4.3 and Lemma A.4.4.

Clearly $\mathscr{P}$ is linear in parameters and hence, convex. Gradient of function $g \in \mathscr{G}$ is given by $\frac{2p_0}{(p+p_0)^2}$ which in turn is bounded by $\frac{2}{p_0}$. Under assumptions of Condition 2.5.1, the functions in $\mathscr{G}$ are Lipschitz with constant $2/\tau$. We can bound the bracketing entropy $H_{2,B}(\delta, \mathscr{G}, P)$ using Lemma A.4.4 and Lemma A.4.3 as

$$H_{2,B}(\delta, \mathscr{G}, P) \leqslant H_\infty(\delta, \mathscr{G}) \leqslant K_1 \left(\frac{1}{\delta\tau}\right)^{k/2},$$

[2] Note that we use the term *density* loosely here for convenience. The actual density is $\langle f(x), w\rangle \cdot p_s(x)$ but we can ignore $p_s(x)$ because it does not depend on our parameters.

for some constant K_1 that depends on k.

On the other hand, for cases where p_0 can be arbitrarily close to zero, i.e., Condition 2.5.1 doesn't hold true, we define $\tau(\delta)$ and $\mathscr{G}_\tau$ as

$$\tau(\delta) = \sup\left\{\tau \geqslant 0 \mid \int_{p_0 \leqslant \tau} p_0 dx \leqslant \delta^2\right\}\,, \tag{A.6}$$

$$\mathscr{G}_\tau = \left\{\frac{2p}{p+p_0}\mathbb{I}p_0 \geqslant \tau \mid p \in \mathscr{P}\right\}\,.$$

Using triangle inequality, for any $g_1, g_2 \in \mathscr{G}$, we have

$$\begin{aligned}\int \|g_1 - g_2\|^2 dx &\leqslant \int \|g_1 - g_2\|^2 \mathbb{I}p_0 \leqslant \tau dx + \int \|g_1 - g_2\|^2 \mathbb{I}p_0 \geqslant \tau dx \\ &\leqslant 2\int \mathbb{I}p_0 \leqslant \tau dx + \int \|g_1 - g_2\|^2 \mathbb{I}p_0 \geqslant \tau dx\,. \end{aligned} \tag{A.7}$$

Assume $\tau(\delta)$ such that (A.6) is satisfied. Using (A.7), we have

$$H_{2,B}(\delta, \mathscr{G}, P) \leqslant H_{2,B}(\sqrt{3}\delta, \mathscr{G}_{\tau(\delta)}, P)\,.$$

Thus, for the cases where p_0 can be arbitrarily close to zero, instead of bounding $H_{2,B}(\delta, \mathscr{G}, P)$, we we bound $H_B(\delta, \mathscr{G}_{\tau(\delta)}, P)$. For any $\delta > 0$, there is a compact subset $K_\delta \in \mathcal{X}$, such that $p_s(X \backslash K_\delta) < \delta$. Using arguments similar to above, function $g \in \mathscr{G}_{\tau(\delta)}$ is Lipschitz with constant $2/\tau(\delta) > 0$. Again using Lemma A.4.4 and Lemma A.4.3, we conclude

$$H_{2,B}(2\delta, \mathscr{G}_{\tau(\delta)}, P) \leqslant H_\infty(\delta, \mathscr{G}_{\tau(\delta)}) \leqslant K_2\left(\frac{1}{\delta\tau(\delta)}\right)^k\,,$$

for some constant K_2 that depends on k. Finally, we use Lemma A.4.5 to conclude $h(\widehat{p}_n, p_0) \rightarrow_{\text{a.s.}} 0$. Further, as $\text{TV}(\widehat{p}_n, p_0) \leqslant h(\widehat{p}_n, p_0)$, we have $h(\widehat{p}_n, p_0) \rightarrow_{\text{a.s.}} 0$ implies $\text{TV}(\widehat{p}_n, p_0) \rightarrow_{\text{a.s.}} 0$. Further

$$\begin{aligned}\|\widehat{w}_f - w^*\|^2 &\leqslant \frac{1}{\lambda_{\min}}\int \left|f(x)^T(\widehat{w}_f - w^*)\right|^2 p_s(x)dx \\ &\leqslant \frac{\sup_x\left\{\left|f(x)^T(\widehat{w}_f - w^*)\right|\right\}}{\lambda_{\min}} \underbrace{\int \left|f(x)^T(\widehat{w}_f - w^*)\right| p_s(x)dx}_{\text{TV}(\widehat{p}_n, p_0)}\,, \end{aligned} \tag{A.8}$$

where $\lambda_{\min}$ is the minimum eigenvalue of covariance matrix $\left[\int f(x)f(x)^T p_s(x)dx\right]$. Note using Proposition 2.4.4, we have $\lambda_{\min} > 0$. Thus, we conclude $\|\widehat{w}_f - w^*\| \rightarrow_{\text{a.s.}} 0$. □

Example 1. Consider a mixture of two Gaussians with $p_s(x|y = 0) := \mathcal{N}(\mu, 1)$ and $p_s(x|y = 1) := \mathcal{N}(-\mu, 1)$. We suppose that the source mixing coefficients are both $\frac{1}{2}$, while the target mixing coefficients are $\alpha(\neq \frac{1}{2}), 1-\alpha$. Assume a class of probabilistic threshold classifiers: $f(x) = [1-c, c]$ for $x \geqslant 0$, otherwise $f(x) = [c, 1-c]$ with $c \in [0,1]$.

Then the population error of MLLS is given by

$$4\left|\frac{(1-2\alpha)(p_s(x \geqslant 0|y=0) - c)}{1-2c}\right|,$$

which is zero only if $c = p_s(x \geqslant 0|y=0)$ for a non-degenerate classifier.

Proof. The intuition behind the construction is, for such an Example, we can get a closed form solution for the population MLLS and hence allows a careful analysis of the estimation error. The classifier $f(x)$ predicts class 0 with probability c and class 1 with probability $1-c$ for $x \geqslant 0$, and vice-versa for $x < 0$. Using such a classifier, the weight estimator is given by:

$$\begin{aligned}
\widehat{w} &= \arg\min_{w} \mathbb{E}\log\langle f(x), w\rangle \\
&\overset{\text{(i)}}{=} \arg\min_{w_0}\left[\int_{-\infty}^{0} \log((1-c)w_0 + c(2-w_0))p_t(x)dx + \int_{0}^{\infty} \log(cw_0 + (1-c)(2-w_0))p_t(x)dx\right] \\
&\overset{\text{(ii)}}{=} \arg\min_{w_0}\left[\log((1-c)w_0 + c(2-w_0))p_t(x \leqslant 0) + \log(cw_0 + (1-c)(2-w_0))p_t(x \geqslant 0)\right],
\end{aligned}$$

where equality (i) follows from $w_1 = 2 - w_0$ and the predictor function and (ii) follows from the fact that within each integral, the term inside the log is independent of x. Differentiating w.r.t. to w_0, we have:

$$\frac{1-2c}{2c + w_0 - 2cw_0}p_t(x \leqslant 0) + \frac{2c-1}{2cw_0 + 2 - 2c - w_0}p_t(x \geqslant 0) = 0$$

$$\frac{1}{2c + w_0 - 2cw_0}p_t(x \leqslant 0) + \frac{-1}{2cw_0 + 2 - 2c - w_0}(1 - p_t(x \leqslant 0)) = 0$$

$$(2cw_0 + 2 - 2c - w_0)p_t(x \leqslant 0) - (2c + w_0 - 2cw_0)(1 - p_t(x \leqslant 0)) = 0$$

$$2p_t(x \leqslant 0) - 2c - w_0 + 2cw_0 = 0\,,$$

which gives $w_0 = \frac{2p_t(x \leqslant 0) - 2c}{1-2c}$. Thus for the population MLLS estimate, the estimation error is given by

$$\|\widehat{w} - w^*\| = 2|w_0 - 2\alpha| = 4\left|\frac{(1-2\alpha)(p_s(x \geqslant 0|y=0) - c)}{1-2c}\right|.$$

□

A.5 Proofs from Section 2.5

The gradient of the MLLS objective can be written as

$$\nabla_w \mathcal{L}(w, f) = \mathbb{E}_t \left[\frac{f(x)}{f(x)^T w} \right] , \tag{A.9}$$

and the Hessian is

$$\nabla_w^2 \mathcal{L}(w, f) = -\mathbb{E}_t \left[\frac{f(x) f(x)^T}{\left(f(x)^T w\right)^2} \right] . \tag{A.10}$$

We use $\lambda_{\min}(X)$ to denote the minimum eigenvalue of the matrix X.

Lemma A.5.1 (Theorem 5.1.1 (Tropp et al., 2015))**.** *Let $X_1, X_2, \ldots, X_n$ be a finite sequence of identically distributed independent, random, symmetric matrices with common dimension k. Assume $0 \preceq X \preceq R \cdot I$ and $\mu_{\min} I \preceq \mathbb{E}X \preceq \mu_{\max} I$. With probability at least $1 - \delta$,*

$$\lambda_{\min} \left(\frac{1}{n} \sum_{i=1}^{n} X_i \right) \geqslant \mu_{\min} - \sqrt{\frac{2R\mu_{\min} \log(\frac{k}{\delta})}{n}} \, . \tag{A.11}$$

Lemma A.5.2. *For any predictor f that satisfies Condition 2.5.1, we have $\|w_f - \widehat{w}_f\| \leqslant \sigma_{f,w_f}^{-1} \mathcal{O}_p\left(m^{-1/2}\right)$.*

Proof. We present our proof in two steps. Step-1 is the non-probabilistic part, i.e., bounding the error $\|\widehat{w}_f - w_f\|$ in terms of the gradient difference $\|\nabla_w \mathcal{L}(w_f, f) - \nabla_w \mathcal{L}_m(w_f, f)\|$. This step uses Taylor's expansion upto second order terms for empirical log-likelihood around the true w^*. Step-2 involves deriving a concentration on the gradient difference using the Lipschitz property implied by Condition 2.5.1. Combining these two steps along with Lemma A.14 concludes the proof. Now we detail each of these steps.

Step-1. We represent the empirical Negative Log-Likelihood (NLL) function with $\mathcal{L}_m$ by absorbing the negative sign to simplify notation. Using a Taylor expansion, we have

$$\mathcal{L}_m(\widehat{w}_f, f) = \mathcal{L}_m(w_f, f) + \langle \nabla_w \mathcal{L}_m(w_f, f), \widehat{w}_f - w_f \rangle + \frac{1}{2} (\widehat{w}_f - w_f)^T \nabla_w^2 \mathcal{L}_m(\widetilde{w}, f_c) (\widehat{w}_f - w_f) \, ,$$

where $\widetilde{w} \in [\widehat{w}_f, w_f]$. With the assumption $f^T w_f \geqslant \tau$, we have $\nabla_w^2 \mathcal{L}_m(\widetilde{w}, f) \geqslant \frac{\tau^2}{\min p_s(y)^2} \nabla_w^2 \mathcal{L}_m(w_f, f)$. Let $\kappa = \frac{\tau^2}{\min p_s(y)^2}$. Using this we get,

$$\mathcal{L}_m(\widehat{w}_f, f) \geqslant \mathcal{L}_m(w_f, f) + \langle \nabla_w \mathcal{L}_m(w_f, f), \widehat{w}_f - w_f \rangle + \frac{\kappa}{2} (\widehat{w}_f - w_f)^T \nabla_w^2 \mathcal{L}_m(w_f, f) (\widehat{w}_f - w_f)$$

$$\underbrace{\mathcal{L}_m(\widehat{w}_f, f) - \mathcal{L}_m(w_f, f)}_{\text{I}} - \langle \nabla_w \mathcal{L}_m(w_f, f), \widehat{w}_f - w_f \rangle \geqslant \frac{\kappa}{2} (\widehat{w}_f - w_f)^T \nabla_w^2 \mathcal{L}_m(w_f, f) (\widehat{w}_f - w_f) \, ,$$

where term-I is less than zero as $\widehat{w}_f$ is the minimizer of empirical NLL $\mathcal{L}_m(\widehat{w}_f, f)$. Ignoring term-I and re-arranging a few terms we get:

$$-\langle \nabla_w \mathcal{L}_m(w_f, f), \widehat{w}_f - w_f \rangle \geqslant \frac{\kappa}{2} (\widehat{w}_f - w_f)^T \nabla_w^2 \mathcal{L}_m(w_f, f) (\widehat{w}_f - w_f) \, ,$$

With first order optimality on w_f, $\langle \nabla_w \mathcal{L}(w_f, f), \widehat{w}_f - w_f \rangle \geqslant 0$. Plugging in this, we have,

$$\langle \nabla_w \mathcal{L}(w_f, f) - \nabla_w \mathcal{L}_m(w_f, f), \widehat{w}_f - w_f \rangle \geqslant \frac{\kappa}{2} (\widehat{w}_f - w_f)^T \nabla_w^2 \mathcal{L}_m(w_f, f)(\widehat{w}_f - w_f)\,,$$

Using Holder's inequality on the LHS we have,

$$\|\nabla_w \mathcal{L}(w_f, f) - \nabla_w \mathcal{L}_m(w_f, f)\| \, \|\widehat{w}_f - w_f\| \geqslant \frac{\kappa}{2} (\widehat{w}_f - w_f)^T \nabla_w^2 \mathcal{L}_m(w_f, f)(\widehat{w}_f - w_f)\,.$$

Let $\widehat{\sigma}_{f,w_f}$ be the minimum eigenvalue of $\nabla_w^2 \mathcal{L}_m(w^*, f_c)$. Using the fact that $(\widehat{w}_f - w_f)^T \nabla_w^2 \mathcal{L}_m(w_f, f)(\widehat{w}_f - w_f) \geqslant \widehat{\sigma}_{\min} \|\widehat{w}_f - w_f\|^2$, we get,

$$\|\nabla_w \mathcal{L}(w_f, f) - \nabla_w \mathcal{L}_m(w_f, f)\| \geqslant \frac{\kappa \widehat{\sigma}_{f,w_f}}{2} \|\widehat{w}_f - w_f\| \,. \tag{A.12}$$

Step-2. The empirical gradient is $\nabla_w \mathcal{L}_m(w_f, f) = \sum_{i=1}^m \frac{\nabla_w \mathcal{L}_1(x_i, w_f, f)}{m}$ where $\nabla \mathcal{L}_1(x_i, w_f, f) = \left[\frac{f_1(x_i)}{\langle f(x_i), w_f \rangle} \cdots \frac{f_l(x_i)}{\langle f(x_i), w_f \rangle} \cdots \frac{f_k(x_i)}{\langle f(x_i), w_f \rangle}\right]_{(k)}$. With the lower bound τ on $f^T w_f$, we can upper bound the gradient terms as

$$\|\nabla_w \mathcal{L}_1(x, w_f, f)\| \leqslant \frac{\|f\|}{\tau} \leqslant \frac{\|f\|_1}{\tau} \leqslant \frac{1}{\tau}\,.$$

As the gradient terms decompose and are independent, using Hoeffding's inequality we have with probability at least $1 - \frac{\delta}{2}$,

$$\|\nabla_w \mathcal{L}(w_f, f) - \nabla_w \mathcal{L}_m(w_f, f)\| \leqslant \frac{1}{2\tau} \sqrt{\frac{\log(4/\delta)}{m}}\,. \tag{A.13}$$

Let σ_{f,w_f} be the minimum eigenvalue of $\nabla_w^2 \mathcal{L}(w_f, f)$. Using lemma A.5.1, with probability at least $1 - \frac{\delta}{2}$,

$$\frac{\widehat{\sigma}_{f,w_f}}{\sigma_{f,w_f}} \geqslant 1 - \tau \sqrt{\frac{\log(2k/\delta)}{m}}\,. \tag{A.14}$$

Plugging (A.13) and (A.14) in (A.12), and applying a union bound, we conclude that with probability at least $1 - \delta$,

$$\begin{aligned}
\|\widehat{w}_f - w_f\|_2 &\leqslant \frac{1}{\kappa\tau} \Big(\sigma_{f,w_f} - \sigma_{f,w_f} \tau \sqrt{\frac{\log(2k/\delta)}{m}}\Big)^{-1} \Big(\sqrt{\frac{\log(4/\delta)}{m}}\Big) \\
&\leqslant \frac{1}{\kappa\tau} \frac{1}{\sigma_{f,w_f}} \Big(1 + \tau \sqrt{\frac{\log(2k/\delta)}{m}}\Big) \sqrt{\frac{\log(4/\delta)}{m}}\,.
\end{aligned}$$

Neglecting the order m term and letting $c = \frac{1}{\kappa\tau}$, we have

$$\|\widehat{w}_f - w_f\| \leqslant \frac{c}{\sigma_{f,w_f}} \sqrt{\frac{\log(4/\delta)}{m}}\,.$$

□

Lemma A.5.3. *For any predictor f and any calibrated predictor f_c that satisfies Condition 2.5.1, we have $\|w_f - w^*\| \leqslant \sigma^{-1}_{f,w^*} \cdot C \cdot \mathbb{E}_t \left[\|f - f_c\| \right]$, for some constant C.*

If we set $f_c(x) = p_s(y|f(x))$, which is a calibrated predictor (Proposition 2.4.4), we can bound the error in terms of the calibration error of f on the source data [3]*: $\|w_f - w^*\| \leqslant \sigma^{-1}_{f,w^*} \cdot C \cdot \mathcal{E}(f)$.*

Proof. We present our proof in two steps. Note, all calculations are non-probabilistic. Step-1 involves bounding the error $\|w_f - w^*\|$ in terms of the gradient difference $\|\nabla_w \mathcal{L}(w^*, f_c) - \nabla_w \mathcal{L}(w^*, f)\|$. This step uses Taylor's expansion on $\mathcal{L}(w_f, f)$ upto the second orderth term for population log-likelihood around the true w^*. Step-2 involves deriving a bound on the gradient difference in terms of the difference $\|f - f_c\|$ using the Lipschitz property implied by Condition 2.5.1. Further, for a crude calibration choice of $f_c(x) = p_s(\cdot|x)$, the gradient difference can be bounded by miscalibration error. We now detail both of these steps.

Step-1. Similar to Lemma 2.5.2, we represent with $\mathcal{L}$ by absorbing the negative sign to simplify notation. Using the Taylor expansion, we have

$$\mathcal{L}(w_f, f) \geqslant \mathcal{L}(w^*, f) + \langle \nabla_w \mathcal{L}(w^*, f), w_f - w^* \rangle + \frac{1}{2}(w_f - w^*)^T \nabla^2_w \mathcal{L}(\widetilde{w}, f)(w_f - w^*) \,,$$

where $\widetilde{w} \in [w_f, w^*]$. With the assumption $f^T w^* \geqslant \tau$, we have $\nabla^2_w \mathcal{L}(\widetilde{w}, f) \geqslant \frac{\tau^2}{\min p_s(y)^2} \nabla^2_w \mathcal{L}(w^*, f)$. Let $\kappa = \frac{\tau^2}{\min p_s(y)^2}$. Using this we get,

$$\mathcal{L}(w_f, f) \geqslant \mathcal{L}(w^*, f) + \langle \nabla_w \mathcal{L}(w^*, f), w_f - w^* \rangle + \frac{\kappa}{2}(w_f - w^*)^T \nabla^2_w \mathcal{L}(w^*, f)(w_f - w^*)$$

$$\underbrace{\mathcal{L}(w_f, f) - \mathcal{L}(w^*, f)}_{\text{I}} \geqslant \langle \nabla_w \mathcal{L}(w_f, f), w_f - w^* \rangle + \frac{\kappa}{2}(w_f - w^*)^T \nabla^2_w \mathcal{L}(w^*, f)(w_f - w^*) \,,$$

where term-I is less than zero as w_f is the minimizer of NLL $\mathcal{L}(w, f)$. Ignoring that term and re-arranging a few terms we get

$$-\langle \nabla_w \mathcal{L}(w^*, f), w_f - w^* \rangle \geqslant \frac{\kappa}{2}(w_f - w^*)^T \nabla^2_w \mathcal{L}(w^*, f)(w_f - w^*) \,.$$

With first order optimality on w^*, $\langle \nabla_w \mathcal{L}(w^*, f_c), w_f - w^* \rangle \geqslant 0$. Using this we have:

$$\langle \nabla_w \mathcal{L}(w^*, f_c), w_f - w^* \rangle - \langle \nabla_w \mathcal{L}(w^*, f), w_f - w^* \rangle \geqslant \frac{\kappa}{2}(w_f - w^*)^T \nabla^2_w \mathcal{L}(w^*, f)(w_f - w^*) \,,$$

$$\langle \nabla_w \mathcal{L}(w^*, f_c) - \nabla_w \mathcal{L}(w^*, f), w_f - w^* \rangle \geqslant \frac{\kappa}{2}(w_f - w^*)^T \nabla^2_w \mathcal{L}(w^*, f)(w_f - w^*) \,.$$

[3] We present two upper bounds because the second is more interpretable while the first is tighter.

As before, let $\sigma_{f,w}$ be the minimum eigenvalue of $\nabla_w^2 \mathcal{L}(w^*, f)$. Using the fact that $(w_f - w^*)^T \nabla_w^2 \mathcal{L}(w^*, f)(w_f - w^*) \geqslant \sigma_{f,w} \left\| w_f - w^* \right\|^2$, we get

$$\langle \nabla_w \mathcal{L}(w^*, f_c) - \nabla_w \mathcal{L}(w^*, f), w_f - w^* \rangle \geqslant \frac{\kappa \sigma_{f,w}}{2} \left\| w_f - w^* \right\|^2 .$$

Using Holder's inequality on the LHS and re-arranging terms gives

$$\left\| \nabla_w \mathcal{L}(w^*, f_c) - \nabla_w \mathcal{L}(w^*, f) \right\| \geqslant \frac{\kappa \sigma_{f,w}}{2} \left\| w_f - w^* \right\| . \tag{A.15}$$

Step-2. By lower bound assumptions $f_c^T w^* \geqslant \tau$ and $f^T w^* \geqslant \tau$, we have

$$\left\| \nabla_w \mathcal{L}(w^*, f_c) - \nabla \mathcal{L}(w^*, f) \right\| \leqslant \mathbb{E}_t \left[\left\| \nabla \mathcal{L}_1(x, w^*, f_c) - \nabla \mathcal{L}_1(x, w^*, f) \right\| \right] \leqslant \frac{1}{\tau^2} \mathbb{E}_t \left[\left\| f_c(x) - f(x) \right\| \right] , \tag{A.16}$$

where the first inequality is implied by Jensen's inequality and the second is implied by the Lipschitz property of the gradient. Further, we have

$$\begin{aligned} \mathbb{E}_t \left[\left\| f_c(x) - f(x) \right\| \right] &= \mathbb{E}_s \left[\frac{p_t(x)}{p_s(x)} \left\| f_c(x) - f(x) \right\| \right] \\ &\leqslant \mathbb{E}_s \left[\max_y \frac{p_t(y)}{p_s(y)} \left\| f_c(x) - f(x) \right\| \right] \\ &\leqslant \max_y \frac{p_t(y)}{p_s(y)} \mathbb{E}_s \left[\left\| f_c(x) - f(x) \right\| \right] . \end{aligned} \tag{A.17}$$

Combining equations (A.15), (A.16), and (A.17), we have

$$\left\| w_f - w^* \right\| \leqslant \frac{2}{\kappa \sigma_{f,w} \tau^2} \max_y \frac{p_t(y)}{p_s(y)} \mathbb{E}_s \left[\left\| f_c(x) - f(x) \right\| \right] . \tag{A.18}$$

Further, if we set $f_c(x) = p_s(\cdot | f(x))$, which is a calibrated predictor according to Proposition 2.4.4, we can bound the error on the RHS in terms of the calibration error of f. Moreover, in the label shift estimation problem, we have the assumption that $p_s(y) \geqslant c > 0$ for all y. Hence, we have $\max_y p_t(y)/p_s(y) \leqslant 1/c$. Using Jensen's inequality, we get

$$\mathbb{E}_s \left\| f_c(x) - f(x) \right\| \leqslant \left(\mathbb{E}_s \left\| f_c(x) - f(x) \right\|^2 \right)^{\frac{1}{2}} = \mathcal{E}(f) . \tag{A.19}$$

Plugging (A.19) back in (A.18),we get the required upper bound. □

Proposition 3. *For any $w \in \mathcal{W}$, we have $\sigma_{f,w} \geqslant p_{s,\min} \sigma_f$ where σ_f is the minimum eigenvalue of $\mathbb{E}_t \left[f(x) f(x)^T \right]$ and $p_{s,\min} = \min_{y \in \mathcal{Y}} p_s(y)$. Furthermore, if f satisfies Condition 2.5.1, we have $p_{s,\min}^2 \cdot \sigma_f \leqslant \sigma_{f,w} \leqslant \tau^{-2} \cdot \sigma_f$, for $w \in \{w_f, w^*\}$.*

Proof. For any $v \in \mathbb{R}^k$, we have

$$v^T \left(-\nabla_w^2 \mathcal{L}(w, f)\right) v = \mathbb{E}_t \left[\frac{\left(v^T f(x)\right)^2}{\left(f(x)^T w\right)^2}\right] \in \left[\frac{1}{a^2}, \frac{1}{b^2}\right] \cdot v^T \mathbb{E}_t \left[f(x) f(x)^T\right] v\,,$$

where

$$a = \max_{x: p_s(x)>0} f(x)^T w \leqslant \frac{1}{p_{s,\min}}$$

and

$$b = \min_{x: p_s(x)>0} f(x)^T w \geqslant \tau$$

if f satisfies Condition 2.5.1 and $w \in \{w_f, w^*\}$. Therefore, we have

$$p_{s,\min}^2 \cdot \sigma_f \leqslant \sigma_{f,w} \leqslant \tau^{-2} \cdot \sigma_f$$

for $w \in \{w_f, w^*\}$.

□

Lemma A.5.4. *Let $f = g \circ \widehat{f}$ be the predictor after post-hoc calibration with squared loss l and g belongs to a function class $\mathcal{G}$ that satisfies the standard regularity conditions, we have*

$$\mathcal{E}(f) \leqslant \min_{g \in \mathcal{G}} \mathcal{E}(g \circ \widehat{f}) + \mathcal{O}_p\left(n^{-1/2}\right)\,. \tag{2.8}$$

Proof. Assume regularity conditions on the model class $\mathcal{G}_\theta$ (injectivity, Lipschitz-continuity, twice differentiability, non-singular Hessian, and consistency) as in Theorem 5.23 of Stein (1981) hold true. Using the injectivity property of the model class as in Kumar et al. (2019), we have for all $g_1, g_2 \in \mathcal{G}$,

$$\mathrm{MSE} g_1 - \mathrm{MSE} g_2 = \mathcal{E}(g_1)^2 - \mathcal{E}(g_2)^2\,. \tag{A.20}$$

Let $\widehat{g}, g^* \in \mathcal{G}$ be models parameterized by $\widehat{\theta}$ and θ^*, respectively. Using the strong convexity of the empirical mean squared error we have,

$$\mathrm{MSE}_n(\widehat{g}) \geqslant \mathrm{MSE}_n(g^*) + \langle \nabla_\theta \mathrm{MSE}_n(g^*), \widehat{\theta} - \theta^* \rangle + \frac{\mu^2}{2} \left\|\widehat{\theta} - \theta^*\right\|_2^2\,,$$

where μ is the parameter constant for strong convexity. Re-arranging a few terms, we have

$$\underbrace{\mathrm{MSE}_n(\widehat{g}) - \mathrm{MSE}_n(g^*)}_{\mathrm{I}} - \langle \nabla_\theta \mathrm{MSE}_n(g^*), \widehat{\theta} - \theta^* \rangle \geqslant \frac{\mu^2}{2} \|\widehat{\theta} - \theta^*\|_2^2\,,$$

where term-I is less than zero because $\widehat{g}$ is the empirical minimizer of the mean-squared error. Ignoring term-I, we get:

$$\frac{\mu^2}{2} \|\widehat{\theta} - \theta^*\|_2^2 \leqslant -\langle \nabla_\theta \mathrm{MSE}_n(g^*), \widehat{\theta} - \theta^* \rangle \leqslant \|\nabla_\theta \mathrm{MSE}_n(g^*)\| \left\|\widehat{\theta} - \theta^*\right\|\,.$$

As the assumed model class is Lipschitz w.r.t. θ, the gradient is bounded by Lipschitz constant $L = c_1$. $\mathbb{E}\nabla_\theta \text{MSE}_n(g^*) = 0$ as g^* is the population minimizer. Using Hoeffding's bound for bounded functions, we have with probability at least $1-\delta$,

$$\|\widehat{\theta} - \theta^*\|_2 \leqslant \frac{c_1}{\mu^2}\sqrt{\frac{\log(2/\delta)}{n}}\,. \tag{A.21}$$

Using the smoothness of the MSEg, we have

$$\text{MSE}\widehat{g} - \text{MSE}g^* \leqslant c_2\|\widehat{\theta} - \theta^*\|_2^2\,, \tag{A.22}$$

where c_2 is the operator norm of the $\nabla^2\text{MSE}g^*$. Combining (A.20), (A.21), and (A.22), we have for some universal constant $c = \frac{c_1 c_2}{\mu^2}$ with probability at least $1-\delta$,

$$\mathcal{E}(\widehat{g})^2 - \mathcal{E}(g^*)^2 \leqslant c\frac{\log(2/\delta)}{n}\,.$$

□

Moreover, with Lemma 2.5.3, depending on the degree of the miscalibration and the method involved to calibrate, we can bound the $\mathcal{E}(f)$. For example, if using vector scaling on a held out training data for calibration, we can use Lemma 2.5.5 to bound the calibration error $\mathcal{E}(f)$, i.e., with probability at least $1-\delta$, we have

$$\mathcal{E}(f) \leqslant \sqrt{\min_{g\in\mathcal{G}} \mathcal{E}(g\circ f)^2 + c\frac{\log(2/\delta)}{n}} \leqslant \min_{g\in\mathcal{G}} \mathcal{E}(g\circ f) + \sqrt{c\frac{\log(2/\delta)}{n}}\,. \tag{A.23}$$

Plugging (A.19) and (A.23) into (A.18), we have with probability at least $1-\delta$ that

$$\|w_f - w^*\| \leqslant \frac{1}{\kappa\sigma_{f,w}\tau^2}\left(\|w^*\|_2\left(\sqrt{c\frac{\log(2/\delta)}{n}} + \min_{g\in\mathcal{G}} \mathcal{E}(g\circ f)\right)\right)\,.$$

Appendix B

Appendix: Online label shift: Optimal dynamic regret meets practical algorithms

B.1 Limitations

Our work is based on the label shift assumption which restricts the applicability of our methods to scenarios where this assumption holds. However, as noted in Section 12.1, the problem of adaptation to changing data distribution is intractable without imposing assumptions on the nature of the shift.

Furthermore, as noted in Remark 3.4.2, our methods in the SOLS settings have a memory requirement that scales linearly with time, which may not be feasible in scenarios where memory is limited. This is reminiscent to FTL / FTRL type algorithms from online learning. We leave the task of deriving theoretical guarantees with reduced storage complexity under non-convex losses as an important future direction.

B.2 Related work

Offline total variation denoising The offline problem of Total Variation (TV) denoising constitutes estimating the ground truth under the observation model in Definition 3.2.1 with the caveat that all observations are revealed ahead of time. This problem is well studied in the literature of locally adaptive non-parametric regression (Donoho and Johnstone, 1994a;b; 1998; Guntuboyina et al., 2020; Kim et al., 2009; Mammen and van de Geer, 1997; Sadhanala et al., 2016b; Tibshirani, 2014; van de Geer, 1990; Wang et al., 2016). The optimal total squared error (TSE) rate for estimation is known to be $\widetilde{O}(T^{1/3}V_T^{2/3}+1)$ (Donoho et al., 1990). Estimating sequences of bounded TV has received a lot of attention in literature mainly because of the fact that these sequences exhibit spatially varying degree of smoothness. Most signals of scientific interest are known to contain spatially localised patterns (Johnstone, 2017). This property also makes the task of designing optimal estimators particularly challenging because the estimator has to efficiently detect localised patterns in the ground truth signal and adjust the amount of smoothing to be applied to optimally trade-off bias and variance.

Non-stationary online learning The problem of online regression can be casted into the dynamic regret minimisation framework of online learning. We assume the notations in Definition 3.2.1. In this framework, at each round the learner makes a decision $\widehat{\theta}_t$. Then the learner suffers a squared error loss $\ell_t(\widehat{\theta}_t) = \|z_t - \widehat{\theta}_t\|_2^2$. The gradient of the loss at the point of decision, $\nabla \ell_t(\widehat{\theta}_t) = 2(\widehat{\theta}_t - z_t)$, is revealed to the learner. The expected dynamic regret against the sequence of comparators $\theta_{1:T}$ is given by

$$\begin{aligned} R(\theta_{1:T}) &= \sum_{t=1}^{T} E[\ell_t(\widehat{\theta}_t) - \ell_t(\theta_t)] && \text{(B.1)} \\ &= \sum_{t=1}^{T} E[\|z_t - \widehat{\theta}_t\|_2^2] - E[\|z_t - \theta_t\|_2^2] && \text{(B.2)} \\ &= \sum_{t=1}^{T} E\left[\|\widehat{\theta}_t\|_2^2 - \|\theta_t\|_2^2 - 2z_t^T\widehat{\theta}_t + 2z_t^T\theta_t\right] && \text{(B.3)} \\ &= \sum_{t=1}^{T} E[\|\widehat{\theta}_t - \theta_t\|_2^2], && \text{(B.4)} \end{aligned}$$

where in the last line we used the fact that the noise ϵ_t (see Definition 3.2.1) is zero mean and independent of the online decisions $\widehat{\theta}_t$. Due to this relation, we conclude that any algorithm that can optimally control the dynamic regret with respect to squared error losses $\ell_t(x) = \|z_t - x\|_2^2$ can be directly used to control the TSE from the ground truth sequence $\theta_{1:T}$.

The minimax estimation rate is defined as follows:

$$R^*(T, V_T) = \min_{\widehat{\theta}_{1:T}} \max_{\substack{\theta_{1:T} \\ \sum_{i=2}^{T} \|\theta_i - \theta_{i-1}\|_1 \leqslant V_T}} \sum_{t=1}^{T} E[\|\widehat{\theta}_t - \theta_t\|_2^2] \tag{B.5}$$

Algorithms that can control the dynamic regret with respect to convex losses such as those presented in the works of Baby and Wang (2023); Besbes et al. (2015); Chang and Shahrampour (2021); Chen et al. (2018); Goel and Wierman (2019); Jacobsen and Cutkosky (2022); Jadbabaie et al. (2015); Yang et al. (2016); Zhang et al. (2018a); Zhao and Zhang (2021); Zhao et al. (2020; 2022) can lead to sub-optimal estimation rates of order $O(\sqrt{T(1+V_T)})$.

On the other hand, algorithms presented in Baby and Wang (2019; 2021; 2022); Baby et al. (2021); Daniely et al. (2015); Hazan and Seshadhri (2007); Raj et al. (2020) exploit the curvature of the losses and attain the (near) optimal estimation rate of $\widetilde{O}(T^{1/3}V_T^{2/3}+1)$.

Algorithm	Run-time	Memory
FLH-FTL (Hazan and Seshadhri, 2007)	$O(T^2)$	$O(T^2)$
Aligator (Baby et al., 2021)	$O(T \log T)$	$O(T)$
Arrows (Baby and Wang, 2019)	$O(T \log T)$	$O(1)$

Table B.1: Run-time and memory complexity of various adaptively minimax optimal online regression algorithms (see Definition 3.2.1). For practical purposes, the storage requirement is negligible even for FLH-FTL. For example, with 10 classes and $T = 1000$, the storage requirement of FLH-FTL is only 40KB, which is insignificant compared to the storage capacity of most modern devices.

Online non-parametric regression The task of estimating a sequence of TV bounded sequence from noisy observations can be cast into the online non-parametric regression framework of Rakhlin and Sridharan (2014). Results on online non-parametric regression against reference class of Lipschitz sequences, Sobolev sequences and isotonic sequences can be found in (Gaillard and Gerchinovitz, 2015; Koolen et al., 2015; Kotłowski et al., 2016) respectively. However as noted in Baby and Wang (2019), these classes feature sequences that are more regular than TV bounded sequences. In fact they can be embedded inside a TV bounded sequence class (Sadhanala et al., 2016a) . So the minimax optimality of an algorithm for TV class implies minimax optimality for the smoother sequence classes as well.

B.3 Omitted proofs from Section 3.3

In the next two lemmas, we verify Assumption 2 for some important loss functions.

Lemma B.3.1 (cross-entropy loss). *Consider a sample $(x, y) \sim Q$. Let $p \in \mathbb{R}_+^K$ and $\widetilde{p}(x) \in \Delta_K$ be a distribution that assigns a weight proportional $\frac{p(i)}{q_0(i)} f_0(i|x)$ to the label i . Let $\ell(\widetilde{p}(x), y) = \sum_{i=1}^K \mathbb{I}\{y = i\} \log(1/p(x)[i])$ be the cross-entropy loss. Let $L(p) := E_{(x,y)\sim Q}[\ell(p(x), y)]$ be its population analogue. Then $L(p)$ is $2\sqrt{K}/\mu$ Lipschitz in $\|\cdot\|_2$ norm over the clipped box $\mathcal{D} := \{p \in \mathbb{R}_+^K : \mu \leqslant p(i) \leqslant 1 \ \forall i \in [K]\}$ which is compact and convex. Further, the true marginals $q_t \in \mathcal{D}$ whenever $q_t(i) \geqslant \mu$ for all $i \in [K]$.*

Proof. We have

$$L(p) = -\sum_{i=1}^K E[E[Q_t(i|x) \log(\widetilde{p}(x)[i])|x]] \tag{B.6}$$

$$= E[\log(\sum_{i=1}^K w_i p(i))] - \sum_{i=1}^K E[E[Q_t(i|x) \log(w_i p(i))|x]], \tag{B.7}$$

where we define $w_i := f_0(i|x)/q_0(i)$. Then we can see that

$$\nabla L(p)[i] = E\left[\frac{w_i}{\sum_{j=1}^K w_j p(j))}\right] - E\left[\frac{Q_t(i|x)}{p(i)}\right]. \tag{B.8}$$

So if $\min_i p(i) \geqslant \mu$, we have that $\frac{w_i}{\sum_{j=1}^K w_j p(j)} \leqslant 1/\mu$ and $Q_t(i|x)/p(i) \leqslant 1/\mu$. So by triangle inequality, $|\nabla L(p)[i]| \leqslant 1/\mu + 1/\mu$.

□

Lemma B.3.2 (binary 0-1 loss)**.** *Consider a sample $(x, y) \sim Q$. Let $p \in \mathbb{R}_+^K$ and $\widetilde{p}(x) \in \Delta_K$ be a distribution that assigns a weight proportional $\frac{p(i)}{q_0(i)} f_0(i|x)$ to the label i. Let $\widehat{y}(x)$ be a sample obtained from the distribution $\widetilde{p}(x)$. Consider the binary 0-1 loss $\ell(\widehat{y}(x), y) = \mathbb{I}(\widehat{y}(x) \neq y)$. Let $L(p) := E_{(x,y)\sim Q, \widehat{y}(x)\sim\widetilde{p}(x)} I(\widehat{y}(x) \neq y)$ be its population analogue. Let $q_0(i) \geqslant \alpha > 0$. Then $L(p)$ is $2K^{3/2}/(\alpha\tau)$ Lipschitz in $\|\cdot\|_2$ norm over the domain $\mathcal{D} := \{p \in \mathbb{R}_+^K : \sum_{i=1}^K p(i) f_0(i|x) \geqslant \tau,\ p(i) \leqslant 1\ \forall i \in [K]\}$ which is compact and convex. Further, the true marginals $q_t \in \mathcal{D}$ whenever $q_t(i) \geqslant \mu$ for all $i \in [K]$.*

Proof. We have that

$$L(p) = \sum_{i=1}^K E[Q(y \neq i|x)\widetilde{p}(x)[i]]. \tag{B.9}$$

Denote $\widetilde{p}(x)[i] = p(i)w_i / \sum_{j=1}^K p(j)w_j$ with $w_j := f_0(i|x)/q_0(i)$. Then we see that

$$\begin{aligned} \left|\frac{\partial \widetilde{p}(x)[i]}{\partial p(i)}\right| &= \left| \frac{w_i}{\sum_{j=1}^K w_j p(j)} - \frac{(w_i p(i)) w_i}{\left(\sum_{j=1}^K w_j p(j)\right)^2} \right| && \text{(B.10)} \\ &\leqslant \frac{1}{\alpha\tau} + \frac{w_i}{\sum_{j=1}^K w_j p(j)} && \text{(B.11)} \\ &\leqslant 2/(\alpha\tau). && \text{(B.12)} \end{aligned}$$

Similarly,

$$\begin{aligned} \left|\frac{\partial \widetilde{p}(x)[i]}{\partial p(j)}\right| &= \frac{w_i p(i) w_j}{\left(\sum_{j=1}^K w_j p(j)\right)^2} && \text{(B.13)} \\ &\leqslant 1/(\alpha\tau). && \text{(B.14)} \end{aligned}$$

Thus we conclude that $\|\nabla \widetilde{p}(x)[i]\|_2 \leqslant 2\sqrt{K}/(\alpha\tau)$, where the gradient is taken with respect to $p \in \mathbb{R}_+^K$.

Therefore,

$$\begin{aligned} \|\nabla L(p)\|_2 &\leqslant \sum_{i=1}^K \|\nabla \widetilde{p}(x)[i]\|_2 && \text{(B.15)} \\ &\leqslant 2K^{3/2}/(\alpha\tau). && \text{(B.16)} \end{aligned}$$

□

Remark B.3.1. *The condition $\sum_{i=1}^K f_0(i|x)p(i) \geqslant \tau$ is closely related to Condition 1 of Garg et al. (2020a). Note that this is strictly weaker than imposing the restriction that the distribution $p(i) \geqslant \mu$ for each i.*

Remark B.3.2. *We emphasize that the conditions in Lemmas B.3.1 and B.3.2 are only sufficient conditions that imply bounded gradients. However, they are not necessary for satisfying bounded gradients property.*

Lemma B.3.3. *Let $\mu, \nu \in \Delta_K$ be such that $\mu[i] = q_t(i)$. Let $s_t = C^{-1} f_0(x_t)$, where C is the confusion matrix defined in Assumption 1. We have that $E[s_t] = \mu$ and $Var(s_t) \leqslant 1/\sigma^2_{min}(C)$*

Proof. Let $\widetilde{q}_t(\widehat{y}_t) = E_{x_t \sim Q_t^X, \widehat{y}(x_t) \sim f_0(x_t)} \mathbb{I}\{\widehat{y}(x_t) = \widehat{y}_t\}$ be the probability that the classifier f_0 predicts the label $\widehat{y}_t$. Here $Q_t^X(x) := \sum_{i=1}^K Q_t(x,i)$. Let's denote $Q_t(\widehat{y}(x_t) = \widehat{y}_t | y_t = i) := E_{x_t \sim Q_t(\cdot|y=i), \widehat{y}(x_t) \sim f_0(x_t)} \mathbb{I}\{\widehat{y}(x_t) = \widehat{y}_t\}$. By law of total probability, we have that

$$\widetilde{q}_t(\widehat{y}_t) = \sum_{i=1}^K Q_t(\widehat{y}(x_t) = \widehat{y}_t | y_t = i) q_t(i) \tag{B.17}$$

$$= \sum_{i=1}^K Q_0(\widehat{y}(x_t) = \widehat{y}_t | y_t = i) q_t(i), \tag{B.18}$$

where the last line follows by the label shift assumption.

Let $\mu, \nu \in \mathbb{R}^K$ be such that $\mu[i] = q_t(i)$ and $\nu[i] = \widetilde{q}_t(i)$. Then the above equation can be represented as $\nu = C\mu$. Thus $\mu = C^{-1}\nu$.

Given a sample $x_t \in Q_t$, the vector $f_0(x_t)$ forms an unbiased estimate of ν. Hence we have that the vector $\widehat{\mu} := C^{-1} f_0(x_t)$ is an unbiased estimate of μ. Moreover,

$$\|\widehat{\mu}\|_2 \leqslant \|C^{-1}\|_2 \|f_0(x_t)\| \tag{B.19}$$

$$\leqslant 1/\sigma_{min}(C). \tag{B.20}$$

Hence the variance of the estimate $\widehat{\mu}$ is bounded by $1/\sigma^2_{min}(C)$.

□

We have the following performance guarantee for online regression due to Baby et al. (2021).

Proposition B.3.4 (Baby et al. (2021))**.** *Let $s_t = C^{-1} f_0(x_t)$. Let $\widehat{q}_t := ALG(s_{1:t-1})$ be the online estimate of the true label marginal q_t produced by the Aligator algorithm by taking $s_{1:t-1}$ as input at a round t. Then we have that*

$$\sum_{t=1}^T E\left[\|\widehat{q}_t - q_t\|_2^2\right] = \widetilde{O}(K^{1/3} T^{1/3} V_T^{2/3} (1/\sigma^{4/3}_{min}(C)) + K), \tag{B.21}$$

where $V_T := \sum_{t=2}^T \|q_t - q_{t-1}\|_1$. Here $\widetilde{O}$ hides dependencies in absolute constants and poly-logarithmic factors of the horizon. Further this result is attained without prior knowledge of the variation V_T.

By following the arguments in Baby and Wang (2021), a similar statement can be derived also for the FLH-FTL algorithm of Hazan and Seshadhri (2007) (Algorithm 14).

Theorem 3.3.1. *Suppose we run Algorithm 2 with the online regression oracle ALG as FLH-FTL (App. B.6) or Aligator (Baby et al., 2021). Then under Assumptions 1 and 2, we have*

$$E[R_{dynamic}(T)] = \widetilde{O}\left(\frac{K^{1/6}T^{2/3}V_T^{1/3}}{\sigma_{min}^{2/3}(C)} + \frac{\sqrt{KT}}{\sigma_{min}(C)}\right), \tag{3.3}$$

where $V_T := \sum_{t=2}^{T} \|q_t - q_{t-1}\|_1$ and the expectation is taken with respect to randomness in the revealed co-variates. Further, this result is attained without prior knowledge of V_T.

Proof. Owing to our carefully crafted reduction from the problem of online label shift to online regression, the proof can be conducted in just a few lines. Let $\widetilde{q}_t$ be the value of ALG$(s_{1:t-1})$ computed at line 2 of Algorithm 2. Recall that the dynamic regret was defined as:

$$R_{\text{dynamic}}(T) = \sum_{t=1}^{T} L_t(\widehat{q}_t) - L_t(q_t) \leqslant \sum_{t=1}^{T} G\|\widehat{q}_t - q_t\|_2 \tag{B.22}$$

Continuing from Eq.(B.22), we have

$$\begin{aligned}
E[R_{\text{dynamic}}(T)] &\leqslant \sum_{t=1}^{T} G \cdot E[\|\widehat{q}_t - q_t\|_2] && \text{(B.23)}\\
&\leqslant \sum_{t=1}^{T} G \cdot E[\|\widetilde{q}_t - q_t\|_2] && \text{(B.24)}\\
&\leqslant \sum_{t=1}^{T} G\sqrt{E\|\widetilde{q}_t - q_t\|_2^2} && \text{(B.25)}\\
&\leqslant G\sqrt{T\sum_{t=1}^{T} E[\|\widetilde{q}_t - q_t\|_2^2]} && \text{(B.26)}\\
&= \widetilde{O}\left(K^{1/6}T^{2/3}V_T^{1/3}(1/\sigma_{min}^{2/3}(C)) + \sqrt{KT}/\sigma_{min}(C)\right), && \text{(B.27)}
\end{aligned}$$

where the second line is due to non-expansivity of projection, the third line is due to Jensen's inequality, fourth line by Cauchy-Schwartz and last line by Proposition B.3.4. This finishes the proof.

□

Next, we provide matching lower bounds (modulo log factors) for the regret in the unsupervised label shift setting. We start from an information-theoretic result which will play a central role in our lower bound proofs.

Proposition B.3.5 (Theorem 2.2 in Tsybakov (2008))**.** *Let $\mathbb{P}$ and $\mathbb{Q}$ be two probability distributions on $\mathcal{H}$, such that $KL(\mathbb{P}||\mathbb{Q}) \leqslant \beta < \infty$, Then for any $\mathcal{H}$-measurable real function $\phi : \mathcal{H} \to \{0, 1\}$,*

$$\max\{\mathbb{P}(\phi = 1), \mathbb{Q}(\phi = 0)\} \geqslant \frac{1}{4}\exp(-\beta). \tag{B.28}$$

Theorem 3.3.2. *Let $V_T \leqslant 64T$. There exists a loss function, a domain $\mathcal{D}$ (in Assumption 2), and a choice of adversarial strategy for generating the data such that for any algorithm, we have $\sum_{t=1}^{T} E([L_t(\widehat{q}_t)] - L_t(q_t)) = \Omega\left(\max\{T^{2/3}V_T^{1/3}, \sqrt{T}\}\right)$, where $\widehat{q}_t \in \mathcal{D}$ is the weight estimated by the algorithm and $q_t \in \mathcal{D}$ is the label marginal at round t chosen by the adversary. Here the expectation is taken with respect to the randomness in the algorithm and the adversary.*

Proof. We start with a simple observation about KL divergence. Consider distributions with density $P(x, y) = P_0(x|y)p(y)$ and $Q(x, y) = P_0(x|y)q(y)$ where $(x, y) \in \mathbb{R} \times [K]$. Note that these distributions are consistent with the label shift assumption. We note that

$$\mathrm{KL}(P||Q) = \sum_{i=1}^{K} \int_{\mathbb{R}} P_0(x|i)p(i) \log\left(\frac{P_0(x|i)p(i)}{P_0(x|i)q(i)}\right) dx \tag{B.29}$$

$$= \sum_{i=1}^{K} \int_{\mathbb{R}} P_0(x|i)p(i) \log\left(\frac{p(i)}{q(i)}\right) dx \tag{B.30}$$

$$= \sum_{i=1}^{K} p(i) \log\left(\frac{p(i)}{q(i)}\right) \tag{B.31}$$

Thus we see that under the label shift assumption, the KL divergence is equal to the KL divergence between the marginals of the labels.

Next, we define a problem instance and an adversarial strategy. We focus on a binary classification problem where the labels is either 0 or 1. As noted before, the KL divergence only depends on the marginal distribution of labels. So we fix the density $Q_0(x|y)$ to be any density such that under the uniform label marginals ($q_0(1) = q_0(0) = 1/2$) we can find a classifier with invertible confusion matrix (recall from Fig. 1 that Q_0 corresponds to the data distribution of the training data set).

Divide the entire time horizon T is divided into batches of size Δ. So there are $M := T/\Delta$ batches (we assume divisibility). Let $\Theta = \left\{\frac{1}{2} - \delta, \frac{1}{2} + \delta\right\}$ be a set of success probabilities, where each probability can define a Bernoulli trial. Here $\delta \in (0, 1/4)$ which will be tuned later.

The problem instance is defined as follows:

- For batch $i \in [M]$, adversary selects a probability $\mathring{q}_i \in \Theta$ uniformly at random.

- For any round t that belongs to the i^{th} batch, sample a label $y_t \sim \text{Ber}(q_t)$ and co-variate $x_t \sim Q_0(\cdot|y_t)$. Here $q_t = \mathring{q}_i$. The co-variate x_t is revealed.
- Let $\widehat{q}_t$ be any estimate of q_t at round t. Define the loss as $L_t(\widehat{q}_t) := \mathbb{I}\{q_t \geqslant 1/2\}(1 - \widehat{q}_t) + \mathbb{I}\{q_t < 1/2\}\widehat{q}_t$.

We take the domain $\mathcal{D}$ in Assumption 2 as $[1/2 - \delta, 1/2 + \delta]$. It is easy to verify that $L_t(\widehat{q}_t)$ is Lipschitz over $\mathcal{D}$. Note that unlike Besbes et al. (2015), we do not have an unbiased estimate of the gradient of loss functions.

Let's compute an upperbound on the total variation incurred by the true marginals. We have

$$\sum_{t=2}^{T} |q_t - q_{t-1}| = \sum_{i=2}^{M} |\mathring{q}_i - \mathring{q}_{i-1}| \tag{B.32}$$

$$\leqslant 2\delta M \tag{B.33}$$

$$\leqslant V_T, \tag{B.34}$$

where the last line is obtained by choosing $\delta = V_T/(2M) = V_T\Delta/(2T)$.

Since at the beginning of each batch, the sampling probability is chosen uniformly at random, the loss function in the current batch is independent of the history available at the beginning of the batch. So only the data in the current batch alone is informative in minimising the loss function in that batch. Hence it is sufficient to consider algorithms that only use the data within a batch alone to make predictions at rounds that falls within that batch.

Now we proceed to bound the regret incurred within batch 1. The computation is identical for any other batches.

Let $\mathbb{P}$ be the joint probability distribution in which labels $(y_1, \ldots, y_\Delta)$ within batch 1 are sampled with success probability $1/2 - \delta$ (i.e $q_t = 1/2 - \delta$)

$$\mathbb{P}(y_1, \ldots, y_\Delta) = \Pi_{i=1}^{\Delta}(1/2 - \delta)^{y_i}(1/2 + \delta)^{1-y_i}. \tag{B.35}$$

Define an alternate distribution $\mathbb{Q}$ such that

$$\mathbb{Q}(y_1, \ldots, y_\Delta) = \Pi_{i=1}^{\Delta}(1/2 + \delta)^{y_i}(1/2 - \delta)^{1-y_i}. \tag{B.36}$$

According to the above distribution the data are independently sampled from Bernoulli trials with success probability $1/2 + \delta$. (i.e $q_t = 1/2 + \delta$)

Moving forward, we will show that by tuning Δ appropriately, any algorithm won't be able to detect between these two alternate worlds with constant probability resulting in sufficiently large regret.

We first bound the KL distance between these two distributions. Let

$$\text{KL}(1/2 - \delta || 1/2 + \delta) := (1/2 + \delta)\log\left(\frac{1/2 + \delta}{1/2 - \delta}\right) + (1/2 - \delta)\log\left(\frac{1/2 - \delta}{1/2 + \delta}\right) \tag{B.37}$$

$$\leqslant_{(a)} (1/2+\delta)\frac{2\delta}{1/2+\delta} - (1/2-\delta)\frac{2\delta}{1/2+\delta} \tag{B.38}$$

$$= \frac{16\delta^2}{1-4\delta^2} \tag{B.39}$$

$$\leqslant_{(b)} \frac{64\delta^2}{3}, \tag{B.40}$$

where in line (a) we used the fact that $\log(1+x) \leqslant x$ for $x > -1$ and observed that $-4\delta/(1+2\delta) > -1$ as $\delta \in (0, 1/4)$. In line (b) we used $\delta \in (0, 1/4)$.

Since $\mathbb{P}$ and $\mathbb{Q}$ are product of the marginals due to independence we have that

$$\mathrm{KL}(\mathbb{P}||\mathbb{Q}) = \sum_{t=1}^{\Delta} \mathrm{KL}(1/2-\delta||1/2+\delta) \tag{B.41}$$

$$\leqslant (64\Delta/3)\cdot\delta^2 \tag{B.42}$$

$$= 16/3 \tag{B.43}$$

$$:= \beta, \tag{B.44}$$

where we used the choices $\delta = \Delta V_T/(2T)$ and $\Delta = (T/V_T)^{2/3}$.

Suppose at the beginning of batch, we reveal the entire observations within that batch $y_{1:\Delta}$ to the algorithm. Note that doing so can only make the problem easier than the sequential unsupervised setting. Let $\widehat{q}_t$ be any measurable function of $y_{1:\Delta}$. Define the function $\phi_t := \mathbb{I}\{\widehat{q}_t \geqslant 1/2\}$. Then by Proposition B.3.5, we have that

$$\max\{\mathbb{P}(\phi_t = 1), \mathbb{Q}(\phi_t = 0)\} \geqslant \frac{1}{4}\exp(-\beta), \tag{B.45}$$

where β is as defined in Eq.(B.44).

Notice that if $q_t = 1/2 - \delta$, then $L_t(\widehat{q}_t) \geqslant 1/2$ for any $\widehat{q}_t \geqslant 1/2$. Similarly if $q_t = 1/2+\delta$, we have that $L_t(\widehat{q}_t) \geqslant 1/2$ for any $\widehat{q}_t < 1/2$.

Further note that $L_t(q_t) = 1/2 - \delta$ by construction.

For notational clarity define $L_t^p(x) := x$ and $L_t^q(x) := 1-x$. We can lower-bound the instantaneous regret as:

$$E[L_t(\widehat{q}_t)] - L_t(q_t) =_{(a)} \frac{1}{2}(E_{\mathbb{P}}[L_t^p(\widehat{q}_t)] - L_t^p(1/2-\delta)) + \frac{1}{2}(E_{\mathbb{Q}}[L_t^q(\widehat{q}_t)] - L_t^q(1/2+\delta)) \tag{B.46}$$

$$\geqslant_{(b)} \frac{1}{2}(E_{\mathbb{P}}[L_t^p(\widehat{q}_t)|\widehat{q}_t \geqslant 1/2] - L_t^p(1/2-\delta)\mathbb{P}(\phi_t = 1) \tag{B.47}$$

$$+ \frac{1}{2}(E_{\mathbb{Q}}[L_t^q(\widehat{q}_t)|\widehat{q}_t < 1/2] - L_t^q(1/2+\delta)\mathbb{Q}(\phi_t = 0) \tag{B.48}$$

$$\geqslant_{(c)} \frac{1}{2}\delta\mathbb{P}(\phi_t = 1) + \frac{1}{2}\delta\mathbb{Q}(\phi_t = 0) \tag{B.49}$$

$$\geqslant \delta/2 \max\{\mathbb{P}(\phi_t = 1), \mathbb{Q}(\phi_t = 0)\} \tag{B.50}$$

$$\geqslant_{(d)} \frac{\delta}{8} \exp(-\beta), \tag{B.51}$$

where in line (a) we used the fact the success probability for a batch is selected uniformly at random from Θ. In line (b) we used the fact that $L_t^p(\widehat{q}_t) - L_t^p(1/2 - \delta) \geqslant 0$ since $\widehat{q}_t \in \mathcal{D} = [1/2 - \delta, 1/2 + \delta]$. Similarly term involving L_t^q is also handled. In line (c) we applied $(E_{\mathbb{P}}[L_t^p(\widehat{q}_t)|\widehat{q}_t \geqslant 1/2] - L_t^p(1/2 - \delta)) \geqslant \delta$ since $E_{\mathbb{P}}[L_t^p(\widehat{q}_t)|\widehat{q}_t \geqslant 1/2] \geqslant 1/2$ and $L_t^p(1/2 - \delta) = 1/2 - \delta$. Similar bounding is done for the term involving $E_{\mathbb{Q}}$ as well. In line (d) we used Eq.(B.45).

Thus we get the total expected regret within batch 1 as

$$\sum_{t=1}^{\Delta} E[L_t(\widehat{q}_t)] - L_t(q_t) \geqslant \frac{\delta\Delta}{8} \exp(-\beta) \tag{B.52}$$

The total regret within any batch $i \in [M]$ can be lower bounded using exactly the same arguments as above. Hence summing the total regret across all batches yields

$$\sum_{t=1}^{T} E[L_t(\widehat{q}_t)] - L_t(q_t) \geqslant \frac{T}{\Delta} \cdot \frac{\delta\Delta}{8} \exp(-\beta) \tag{B.53}$$

$$= \frac{V_T \Delta}{16} \cdot \exp(-\beta) \tag{B.54}$$

$$= T^{2/3} V_T^{1/3} \exp(-\beta)/16. \tag{B.55}$$

The $\Omega(\sqrt{T})$ part of the lowerbound follows directly from Theorem 3.2.1 in Hazan (2016) by choosing $\mathcal{D}$ with diameter bounded by $\Omega(1)$.

□

B.4 Design of low switching online regression algorithms

Even-though Algorithm 4 has attractive performance guarantees, it requires retraining with weighted ERM at every round. This is not satisfactory since the retraining can be computationally expensive. In this section, we aim to design a version of Algorithm 4 with few retraining steps while not sacrificing the statistical efficiency (up to constants). To better understand why this goal is attainable, consider a time window $[1, n] \subseteq [T]$ where the true label marginals remain constant or drift very slowly. Due to the slow drift, one reasonable strategy is to re-train the model (with weighted ERM) using the past data only at time points within $[1, n]$ that are powers of 2 (i.e via a doubling epoch schedule). For rounds $t \in [1, n]$ that are not powers of 2, we make predictions with a previous model h_{prev} computed at $t_{\text{prev}} := 2^{\lfloor \log_2 t \rfloor}$ which is trained using data seen upto the time t_{prev}. Observe that this constitutes at least half of the data seen until round t. This observation when

Algorithm 12 `LPA`: a black-box reduction to produce a low-switching online regression algorithm

```
input Online regression oracle ALG, failure probability $\delta$, maximum standard deviation $\sigma$ (see Definition 3.2.1).
Initialize prev $= 0 \in \mathbb{R}^K$, $b = 1$
Get estimate $\widetilde{\theta}_t$ from ALG$(z_{1:t-1})$
Output $\widehat{\theta}_t =$ prev
Receive an observation $z_t$
// test to detect non-staionarity
if $\sum_{j=b+1}^{t} \|\text{prev} - \widetilde{\theta}_j\|_2^2 > 5K\sigma^2 \log(2T/\delta)$ then
    Set $b = t + 1$, prev $= z_t$
    Restart ALG
else if $t - b + 1$ is a power of 2 then
    Set prev $= \nicefrac{\sum_{j=b}^{t} z_j}{t-b+1}$
end if
Update ALG with $z_t$
```

combined with the slow drift of label marginals implies that the performance of the model h_{prev} at round t will be comparable to the performance of a model obtained by retraining using entire data collected until round t.

To formalize this idea, we need an efficient online change-point-detection strategy that can detect intervals where the TV of the *true* label marginals is low and retrain only (modulo at most $\log T$ times within a low TV window) when there is enough evidence for sufficient change in the TV of the true marginals. We address this problem via a two-step approach. In the first step, we construct a generic black-box reduction that takes an online regression oracle as input and converts it into another algorithm with the property that the number of switches in its predictions is controlled without sacrificing the statistical performance. Recall that the purpose of the online regression oracles is to track the true label marginals. The output of our low-switching online algorithm remains the same as long as the TV of the *true* label marginals (TV computed from the time point of the last switch) is sufficiently small. Then we use this low-switching online regression algorithm to re-train the classifier when a switch is detected.

We next provide the **L**ow switching through **P**hased **A**veraging (LPA) (Algorithm 12), our black-box reduction to produce low switching regression oracles. We remark that this algorithm is applicable to the much broader context of *online regression* or *change point detection* and can be of independent interest.

We now describe the intuition behind Algorithm 12. The purpose of Algorithm 12 is to denoise the observations z_t and track the underlying ground truth θ_t in a statistically efficient manner while incurring low switching cost. Hence it is applicable to the broader context of online non-parametric regression (Baby and Wang, 2019; Baby et al., 2021; Raj et al., 2020) and offline non-parametric regression (Tibshirani, 2014; Wang et al., 2015).

Algorithm 12 operates by adaptively detecting low TV intervals. Within each time window it performs a phased averaging in a doubling epoch schedule. i.e consider a low TV window $[b, n]$. For a round $t \in [b, n]$ let $t_{\text{prev}} := 2^{\lfloor \log_2 (t-b+1) \rfloor}$. In round t, the algorithm plays the average of the observations $z_{b:t_{\text{prev}}}$. So we see that in any low TV window, the algorithm changes its output only at-most $O(\log T)$ times.

For the above scheme to not sacrifice statistical efficiency, it is important to efficiently detect windows with low TV of the true label marginals. Observe that the quantity `prev` computes the average of at-least half of the observations within a time window that start at time b. So when the TV of the ground truth within a time window $[b, t]$ is small, we can expect the average to be a good enough representation of the entire ground truth sequence within that time window. Consider the quantity $R_t := \sum_{j=b+1}^{t} \|\text{prev} - \theta_j\|_2^2$ which is the total squared error (TSE) incurred by the fixed decision `prev` within the current time window. Whenever the TV of the ground truth sequence $\theta_{b:t}$ is large, there will be a large bias introduced by `prev` due to averaging. Hence in such a scenario the TSE will also be large indicating non-stationarity. However, we can't compute R_t due to the unavailability of θ_j. So we approximate R_t by replacing θ_j with the estimates $\widetilde{\theta}_j$ coming from the input online regression algorithm that is not constrained by switching cost restrictions. This is the rationale behind the non-stationarity detection test at Step 5. Whenever a non-staionarity is detected we restart the input online regression algorithm as well as the start position for computing averages (in Step 6).

We have the following guarantee for Algorithm 12.

Theorem B.4.1. *Suppose the input black box ALG given to Algorithm 12 is adaptively minimax optimal (see Definition 3.2.1). Then the number of times Algorithm 12 switches its decision is at most $\widetilde{O}(T^{1/3} V_T^{2/3})$ with probability at least $1-\delta$. Further, Algorithm 12 satisfies $\sum_{t=1}^{T} \|\widehat{\theta}_t - \theta_t\|_2^2 = \widetilde{O}(T^{1/3} V_T^{2/3})$ with probability at least $1 - \delta$, where $V_T = \sum_{t=2}^{T} \|\theta_t - \theta_{t-1}\|_1$.*

Remark B.4.1. *Since Algorithm 12 is a black-box reduction, there are a number of possible candidates for the input policy ALG that are adaptively minimax. Examples include FLH with online averages as base learners (Hazan and Seshadhri, 2007) or Aligator algorithm (Baby et al., 2021).*

Armed with a low switching online regression oracle `LPA`, one can now tweak Algorithm 4 to have sparse number of retraining steps while not sacrificing the statistical efficiency (up to multiplicative constants). The resulting procedure is described in Algorithm 13 (in App. B.5) which enjoys similar rates as in Theorem 3.4.1 (see Theorem B.5.3).

B.5 Omitted proofs from Section 3.4

First we recall a result from Baby et al. (2021).

Proposition B.5.1 (Theorem 5 of Baby et al. (2021))**.** *Consider the online regression protocol defined in Definition 3.2.1. Let $\widehat{\theta}_t$ be the estimate of the ground truth produced by the Aligator algorithm from Baby et al. (2021). Then with probability at-least $1 - \delta$, the* *total squared error (TSE) of Aligator satisfies*

$$\sum_{t=1}^{T} \|\theta_t - \widehat{\theta}_t\|_2^2 = \widetilde{O}(T^{1/3} V_T^{2/3} + 1), \tag{B.56}$$

where $V_T = \sum_{t=2}^{T} \|\theta_t - \theta_{t-1}\|_1$. This bound is attained without any prior knowledge of the variation V_T.

The high probability guarantee also implies that

$$\sum_{t=1}^{T} E[\|\theta_t - \widehat{\theta}_t\|_2^2] = \widetilde{O}(T^{1/3} V_T^{2/3} + 1), \tag{B.57}$$

where the expectation is taken with respect to randomness in the observations.

By following the arguments in Baby and Wang (2021), a similar statement can be derived also for the FLH-FTL algorithm of Hazan and Seshadhri (2007) (Algorithm 14).

Next, we verify that the noise condition in Definition 3.2.1 is satisfied for the empirical label marginals computed at Step 5 of Algorithm 4.

Lemma B.5.2. *Let s_t be as in Step 5 of Algorithm 4. Then it holds that $s_t = q_t + \epsilon_t$ with ϵ_t being independent across t and $\mathrm{Var}(\epsilon_t) \leqslant 1/N$.*

Proof. Since s_t is simply the empirical label proportions, it holds that $E[s_t] = q_t$. Further $\mathrm{Var}(s_t) \leqslant 1$ as the indicator function is bounded by $1/N$. This concludes the proof. □

Theorem 3.4.1. *Suppose the true label marginal satisfies $\min_{t,k} q_t(k) \geqslant \mu > 0$. Choose the online regression oracle in Algorithm 4 as FLH-FTL (App. G.3) or Aligator from Baby et al. (2021) with its predictions clipped such that $\widehat{q}_t[k] \geqslant \mu$. Then with probability at least $1-\delta$, Algorithm 4 produces hypotheses with $R^{\mathcal{H}}_{dynamic} = \widetilde{O}\left(T^{2/3} V_T^{1/3} + \sqrt{T \log(|\mathcal{H}|/\delta)}\right)$, where $V_T = \sum_{t=2}^{T} \|q_t - q_{t-1}\|_1$. Further, this result is attained without any prior knowledge of the variation budget V_T.*

Proof. In the proof we first proceed to bound the instantaneous regret at round t. Re-write the population loss as:

$$L_t(h) = \frac{1}{N(t-1)} \sum_{i=1}^{t-1} \sum_{j=1}^{N} E\left[\frac{q_t(y_{ij})}{q_i(y_{ij})} \ell(h(x_{ij}), y_{ij})\right], \tag{B.58}$$

where the expectation is taken with respect to randomness in the samples.

We define the following quantities:

$$L_t^{\mathrm{emp}}(h) := \frac{1}{N(t-1)} \sum_{i=1}^{t-1} \sum_{j=1}^{N} \frac{q_t(y_{ij})}{q_i(y_{ij})} \ell(h(x_{ij}), y_{ij}), \tag{B.59}$$

$$\widetilde{L}_t(h) := \frac{1}{N(t-1)} \sum_{i=1}^{t-1} \sum_{j=1}^{N} E\left[\frac{\widehat{q}_t(y_{ij})}{\widehat{q}_i(y_{ij})} \ell(h(x_{ij}), y_{ij}) \right], \tag{B.60}$$

and

$$\widetilde{L}_t^{\text{emp}}(h) := \frac{1}{N(t-1)} \sum_{i=1}^{t-1} \sum_{j=1}^{N} \frac{\widehat{q}_t(y_{ij})}{\widehat{q}_i(y_{ij})} \ell(h(x_{ij}), y_{ij}). \tag{B.61}$$

We decompose the regret at round t as

$$L_t(h_t) - L_t(h_t^*) = L_t(h_t) - \widetilde{L}_t(h_t) + \widetilde{L}_t(h_t) - \widetilde{L}_t^{\text{emp}}(h_t) + L_t^{\text{emp}}(h_t^*) - L_t(h_t^*) + \widetilde{L}_t^{\text{emp}}(h_t) - L_t^{\text{emp}}(h_t^*) \tag{B.62}$$

$$\leqslant \underbrace{L_t(h_t) - \widetilde{L}_t(h_t)}_{\text{T1}} + \underbrace{\widetilde{L}_t(h_t) - \widetilde{L}_t^{\text{emp}}(h_t)}_{\text{T2}} + \underbrace{L_t^{\text{emp}}(h_t^*) - L_t(h_t^*)}_{\text{T3}} + \underbrace{\widetilde{L}_t^{\text{emp}}(h_t^*) - L_t^{\text{emp}}(h_t^*)}_{\text{T4}}, \tag{B.63}$$

where in the last line we used Eq.(8.1). Now we proceed to bound each terms as note above.

Note that for any label m,

$$\left| \frac{q_t(m)}{q_i(m)} - \frac{\widehat{q}_t(m)}{\widehat{q}_i(m)} \right| \leqslant \left| \frac{q_t(m)}{q_i(m)} - \frac{q_t(m)}{\widehat{q}_i(m)} \right| + \left| \frac{q_t(m)}{\widehat{q}_i(m)} - \frac{\widehat{q}_t(m)}{\widehat{q}_i(m)} \right| \tag{B.64}$$

$$\leqslant \frac{1}{\mu^2} \left(|q_i(m) - \widehat{q}_i(m)| + |q_t(m) - \widehat{q}_t(m)| \right), \tag{B.65}$$

where in the last line, we used the assumption that the minimum label marginals (and hence of the online estimates via clipping) is bounded from below by μ. So by applying triangle inequality and using the fact that the losses are bounded by B in magnitude, we get

$$T1 \leqslant \frac{B}{N(t-1)\mu^2} \sum_{i=1}^{t-1} \sum_{j=1}^{N} E\left[\|\widehat{q}_i - q_i\|_1 + \|\widehat{q}_t - q_t\|_1 \right] \tag{B.66}$$

$$\leqslant \frac{B\sqrt{K}}{(t-1)\mu^2} \sum_{i=1}^{t-1} E\left[\|\widehat{q}_i - q_i\|_2 + \|\widehat{q}_t - q_t\|_2 \right] \tag{B.67}$$

$$\leqslant_{(a)} \frac{B\sqrt{K}}{\mu^2} \left(E[\|\widehat{q}_t - q_t\|_2] + \sqrt{\frac{\sum_{i=1}^{t-1} E[\|q_i - \widehat{q}_i\|_2^2]}{t-1}} \right) \tag{B.68}$$

$$\leqslant_{(b)} \frac{B\sqrt{K}}{\mu^2} \left(E[\|\widehat{q}_t - q_t\|_2] + \phi \cdot \frac{V_T^{1/3}}{(t-1)^{1/3}} \right), \tag{B.69}$$

where line (a) is a consequence of Jensen's inequality. In line (b) we used the following fact: by Lemma B.5.2 and Proposition B.3.4, the expected cumulative error of the online oracle at any step is bounded by $\phi t^{1/3} V_t^{2/3}$ for some multiplier ϕ which can contain poly-logarithmic factors of the horizon (see Proposition B.5.1).

Proceeding in a similar fashion, the term $T4$ can be bounded by Eq.(B.69).

Next, we proceed to handle T3. Let $h \in \mathcal{H}$ be any fixed hypothesis. Then each summand in Eq.(B.59) is an independent random variable assuming values in $[0, B/\mu]$ (recall that the losses lie within $[0, B]$). Hence by Hoeffding's inequality we have that

$$L_t^{\text{emp}}(h) - L_t(h) \leqslant \frac{B}{\mu}\sqrt{\frac{\log(3T|\mathcal{H}|/\delta)}{N(t-1)}}, \tag{B.70}$$

$$\leqslant \frac{B}{\mu}\sqrt{\frac{\log(3T|\mathcal{H}|/\delta)}{(t-1)}}, \tag{B.71}$$

with probability at-least $1 - \delta/(3T|\mathcal{H}|)$. Now taking union bound across all hypotheses in $\mathcal{H}$, we obtain that:

$$T3 \leqslant \frac{B}{\mu}\sqrt{\frac{\log(3|\mathcal{H}|/\delta)}{(t-1)}}, \tag{B.72}$$

with probability at-least $1 - \delta/(3T)$.

To bound T2, we notice that it is not possible to directly apply Hoeffding's inequality because the summands in Eq.(B.60) are correlated through the estimates of the online algorithm. So in the following, we propose a trick to decorrelate them. For any hypothesis $h \in \mathcal{H}$, we have that

$$\frac{\widehat{q}_t(y_{ij})}{\widehat{q}_i(y_{ij})}\ell(h(x_{ij}, y_{ij})) - E\left[\frac{\widehat{q}_t(y_{ij})}{\widehat{q}_i(y_{ij})}\ell(h(x_{ij}, y_{ij}))\right] \tag{B.73}$$

$$= \underbrace{\left(\frac{\widehat{q}_t(y_{ij})}{\widehat{q}_i(y_{ij})} - \frac{q_t(y_{ij})}{q_i(y_{ij})}\right)\ell(h(x_{ij}, y_{ij}))}_{U_{ij}} - \tag{B.74}$$

$$\underbrace{E\left[\left(\frac{\widehat{q}_t(y_{ij})}{\widehat{q}_i(y_{ij})} - \frac{q_t(y_{ij})}{q_i(y_{ij})}\right)\ell(h(x_{ij}, y_{ij}))\right]}_{V_{ij}} + \tag{B.75}$$

$$\underbrace{\frac{q_t(y_{ij})}{q_i(y_ij)}\ell(h(x_{ij}, y_{ij})) - E\left[\frac{q_t(y_{ij})}{q_i(y_ij)}\ell(h(x_{ij}, y_{ij}))\right]}_{W_{ij}}. \tag{B.76}$$

Now using Eq.(B.65) and proceeding similar to the bouding steps of Eq.(B.69), we obtain

$$
\begin{aligned}
\frac{1}{N(t-1)}\sum_{i=1}^{t-1}\sum_{j=1}^{N} U_{ij} &\leqslant \frac{B}{N(t-1)\mu^2}\sum_{i=1}^{t-1}\sum_{j=1}^{N} \|\widehat{q}_i - q_i\|_1 + \|\widehat{q}_t - q_t\|_1 && \text{(B.77)}\\
&\leqslant \frac{B\sqrt{K}}{\mu^2(t-1)}\sum_{i=1}^{t-1} \|\widehat{q}_i - q_i\|_2 + \|\widehat{q}_t - q_t\|_2 && \text{(B.78)}\\
&\leqslant_{(a)} \frac{B\sqrt{K}}{\mu^2}\left(\|\widehat{q}_t - q_t\|_2 + \sqrt{\frac{\sum_{i=1}^{t-1}\|q_i - \widehat{q}_i\|_2^2}{t-1}}\right) && \text{(B.79)}\\
&\leqslant_{(b)} \frac{B\sqrt{K}}{\mu^2}\left(\|\widehat{q}_t - q_t\|_2 + \phi\cdot\frac{V_T^{1/3}}{(t-1)^{1/3}}\right), && \text{(B.80)}
\end{aligned}
$$

with probability at-least $1-\delta/3$. In line (a) we used Jensen's inequlaity and in the last line we used the fact the the online oracle attains a high probability bound on the total squared error (TSE) (see Proposition B.5.1).

$\frac{1}{N(t-1)}\sum_{i=1}^{t-1}\sum_{j=1}^{N} V_{ij}$ can be bounded using the same expression as above using similar logic.

To bound $\frac{1}{N(t-1)}\sum_{i=1}^{t-1}\sum_{j=1}^{N} W_{ij}$, we note that it is the sum of independent random variables. Hence using the same arguments used to obtain Eq.(B.71), we have that

$$
\frac{1}{N(t-1)}\sum_{i=1}^{t-1}\sum_{j=1}^{N} W_{ij} \leqslant \frac{B}{\mu}\sqrt{\frac{\log(3T|\mathcal{H}|/\delta)}{(t-1)}}, \tag{B.81}
$$

with probability at-least $1-\delta/(3T|\mathcal{H}|)$. Hence taking a union bound across all hypothesis classes and across the high probability event of low TSE for the online algorithm yields that

$$
T2 \leqslant \frac{2B\sqrt{K}}{\mu^2}\left(\|\widehat{q}_t - q_t\|_2 + \phi\cdot\frac{V_T^{1/3}}{(t-1)^{1/3}}\right) + \frac{B}{\mu}\sqrt{\frac{\log(3T|\mathcal{H}|/\delta)}{(t-1)}}, \tag{B.82}
$$

with probability at-least $1-2\delta/(3T)$.

Combining the bounds developed for T1,T2,T3 and T4 and by taking a union bound across the event that resulted in Eq.(B.72), we obtain the following bound on instantaneous regret.

$$
L_t(h_t) - L_t(h_t^*) \leqslant \frac{2B\sqrt{K}}{\mu^2}\left(\|\widehat{q}_t - q_t\|_2 + E[|\widehat{q}_t - q_t\|_2] + \phi\cdot\frac{V_T^{1/3}}{(t-1)^{1/3}} + \sqrt{\frac{\log(3T|\mathcal{H}|/\delta)}{(t-1)}}\right), \tag{B.83}
$$

with probability at-least $1 - \delta/T$.

Note that via Jensen's inequality:

$$\sum_{t=1}^{T} E[\|q_t - \widehat{q}_t\|_2] \leqslant \sqrt{T \sum_{t=1}^{T} E[\|q_t - \widehat{q}_t\|_2^2]} \tag{B.84}$$

$$\leqslant \phi T^{2/3} V_T^{1/3}, \tag{B.85}$$

where in the last line we used Proposition B.5.1.

Similarly it can be shown that

$$\sum_{t=1}^{T} \|q_t - \widehat{q}_t\|_2 \leqslant \phi T^{2/3} V_T^{1/3}, \tag{B.86}$$

under the event that resulted in Eq.(B.83).

Observe that

$$\sum_{t=1}^{T} \frac{V_T^{1/3}}{t^{1/3}} \leqslant 2T^{2/3} V_T^{1/3}. \tag{B.87}$$

Finally note that

$$\sum_{t=1}^{T} \frac{1}{\sqrt{t}} \leqslant 2\sqrt{T}. \tag{B.88}$$

Hence combining the above bounds and adding Eq.(B.83) across all time steps, followed by a union bound across all rounds, we obtain that

$$\sum_{t=1}^{T} L_t(h_t) - L_t(h_t^*) \leqslant \frac{4B\sqrt{K}}{\mu^2} \left(3\phi T^{2/3} V_T^{1/3} + \sqrt{T \log(3T|\mathcal{H}|/\delta)}\right), \tag{B.89}$$

with probability at-least $1 - \delta$.

□

Next, we prove Theorem B.4.1.

Theorem B.4.1. *Suppose the input black box ALG given to Algorithm 12 is adaptively minimax optimal (see Definition 3.2.1). Then the number of times Algorithm 12 switches its decision is at most $\widetilde{O}(T^{1/3} V_T^{2/3})$ with probability at least $1 - \delta$. Further, Algorithm 12 satisfies $\sum_{t=1}^{T} \|\widehat{\theta}_t - \theta_t\|_2^2 = \widetilde{O}(T^{1/3} V_T^{2/3})$ with probability at least $1 - \delta$, where $V_T = \sum_{t=2}^{T} \|\theta_t - \theta_{t-1}\|_1$.*

Proof. First we proceed to bound the number of switches. Observe that between two time points where condition in Line 5 of Algorithm 12 evaluates true, we can have at-most $\log T$ switches due to the doubling epoch schedule in Line 8.

We first bound the number of times, condition in Line 5 is satisfied. Suppose for some some time t, we have that $\sum_{j=b+1}^{t} \|\text{prev} - \widetilde{\theta}_j\|_2^2 > 4K\sigma^2 \log(T/\delta)$. Suppose throughout the run of the algorithm, this is i^{th} time the previous condition is satisfied. Let $n_i := t - b + 1$ and let $C_i = \text{TV}[b \to t]$ where $\text{TV}[p \to q] = \sum_{t=p+1}^{q} \|\theta_t - \theta_{t-1}\|_1$. Due to the doubling epoch schedule, we have that that $\text{prev} = \frac{1}{\ell}\sum_{j=b}^{\ell} y_j$ and $E[\text{prev}] = \frac{1}{\ell}\sum_{j=b}^{\ell} \theta_j$ for some $n_i \geqslant \ell \geqslant (t-b+1)/2 = n_i/2$.

So we have

$$\sum_{j=b+1}^{t} \|\text{prev} - \widetilde{\theta}_j\|_2^2 \leqslant \sum_{j=b+1}^{t} 2\|\text{prev} - \theta_j\|_2^2 + 2\|\widetilde{\theta}_j - \theta_j\|_2^2 \tag{B.90}$$

$$\leqslant \sum_{j=b+1}^{t} 2\|E[\text{prev}] - \theta_j\|_2^2 + 2\|\text{prev} - E[\text{prev}]\|_2^2 + 2\|\widetilde{\theta}_j - \theta_j\|_2^2 \tag{B.91}$$

$$\leqslant_{(a)} 2(\ell C_i^2 + 2\sigma^2 K \log(2T/\delta)) + 2\phi n_i^{1/3} C_i^{2/3} \tag{B.92}$$

$$\leqslant 4\max\{n_i C_i^2, \phi n_i^{1/3} C_i^{2/3}\} + 4\sigma^2 K \log(2T/\delta)), \tag{B.93}$$

with probability at-least $1 - \delta/(T)$. In line (a) we used the following facts: i) Due to Hoeffding's inequality, $\|\text{prev} - E[\text{prev}]\|_2^2 \leqslant \sigma^2 K \log(4T/\delta))/\ell \leqslant 2\sigma^2 K \log(2T/\delta))/n_i$ with probability at-least $1-\delta/(2T)$; ii) $\|E[\text{prev}]-\theta_j\|_2 = \|\frac{1}{\ell}\sum_{i=b}^{\ell} \theta_i - \theta_j\|_2 \leqslant \frac{1}{\ell}\sum_{i=b}^{\ell} \|\theta_i - \theta_j\|_2] \leqslant C_i$; iii) $\|\widetilde{\theta}_j - \theta_j\|_2^2 \leqslant \phi n_i^{1/3} C_i^{2/3}$ with probability at-least $1 - \delta/(2T)$ due to condition in Theorem B.4.1; iv) Union bound over the events in (i) and (iii).

Since the condition in Line 5 is satisfied at round t, Eq.(B.93) will imply that $5K\sigma^2 \log(2T/\delta) \leqslant 4\max\{n_i C_i^2, \phi n_i^{1/3} C_i^{2/3}\} + 4\sigma^2 K \log(2T/\delta))$. Rearranging the above, we find that

$$C_i \gtrsim K/\sqrt{n_i}, \tag{B.94}$$

where we suppress the dependence on constants and $\log T$.

Let the condition in Line 5 be satisfied M number of times. By union bound, we have that with probability at-least $1 - \delta$

$$V_T \geqslant \sum_{i=1}^{M} C_i \tag{B.95}$$

$$\gtrsim \sum_{i=1}^{M} K/\sqrt{n_i} \tag{B.96}$$

$$\gtrsim_{(a)} KM \frac{1}{\sqrt{(1/M)\sum_{i=1}^{M} n_i}} \tag{B.97}$$

$$\gtrsim KM^{3/2}/\sqrt{T}, \tag{B.98}$$

where in Line (a) we used Jensen's inequality. Rearranging we get that

$$M = \widetilde{O}(T^{1/3}V_T^{2/3}K^{-2/3}), \tag{B.99}$$

with probability at-least $1 - \delta$.

Now we proceed to bound the total squared error (TSE) incurred by Algorithm 12. Let $\widehat{\theta}_j$ be the output of Algorithm 12 at round j. Suppose at times $b - 1$ and $c + 1$, the condition in Line (5) is satisfied. Observe that the condition in Line 5 is not satisfied for any times in $[b, c]$. Then we can conclude that within the interval $[b, c]$ we have that $\sum_{j=b}^{c} \|\widehat{\theta}_j - \widetilde{\theta}_j\|_2^2 \leqslant 5K\sigma^2 \log(4T/\delta)\log(T)$, since there are only at-most $\log T$ times within $[b, c]$ where condition in Line 9 is satisfied. So we have that

$$\sum_{j=b}^{c} \|\widehat{\theta}_j - \theta_j\|_2^2 \leqslant \sum_{j=b}^{c} \|\widehat{\theta}_j - \widetilde{\theta}_j\|_2^2 + \|\theta_j - \widetilde{\theta}_j\|_2^2 \tag{B.100}$$

$$\leqslant 5K\sigma^2 \log(2T/\delta)\log(T) + \phi \cdot n_i^{1/3} C_i^{2/3}, \tag{B.101}$$

with probability at-least $1 - \delta/T$. Here $n_i := b - c + 1$ and $C_i := \mathrm{TV}[b \to c]$. Further we have that $\|\widehat{\theta}_{c+1} - \theta_{c+1}\|_2^2 \leqslant 2B^2$ due to the boundedness condition in Definition 3.2.1.

Thus overall we have that $\sum_{j=b}^{c+1} = \widetilde{O}(K + n_i^{1/3} C_i^{2/3})$, with probability at-least $1 - \delta$ for any interval [b,c+1] such that condition in Line 5 is satisfied at times $b - 1$ and $c + 1$. Thus we have that

$$\sum_{t=1}^{T} \|\widehat{\theta}_j - \theta_j\|_2^2 \lesssim \sum_{i=1}^{M} K + n_i^{1/3} C_i^{2/3} \tag{B.102}$$

$$\lesssim_{(a)} T^{1/3}V_T^{2/3}K^{1/3} + \sum_{i=1}^{M} n_i^{1/3} C_i^{2/3} \tag{B.103}$$

$$\lesssim_{(b)} T^{1/3}V_T^{2/3}K^{1/3} + \left(\sum_{i=1}^{M} n_i\right)^{1/3} \left(\sum_{i=1}^{M} C_i\right)^{2/3} \tag{B.104}$$

$$\lesssim T^{1/3}V_T^{2/3}K^{1/3}, \tag{B.105}$$

with probability at-least $1 - \delta$. In line (a) we used Eq.(B.99). In line (b) we used Holder's inequality with the dual norm pair $(3, 3/2)$. This concludes the proof.

□

We now present the tweak of Algorithm 4 by instantiating ALG with Algorithm 12 and prove its regret guarantees. The resulting algorithm is described in Algorithm 13.

Algorithm 13 `Lazy-TrainByWeights`: handling label shift with sparse ERM calls

```
Input: Instance ALG of Algorithm 12, A hypothesis Class $\mathcal{H}$
1: At round $t \in [T]$, get estimated label marginal $\widehat{q}_t \in \mathbb{R}^K$ from ALG($s_{1:t-1}$).
2: if $\widehat{q}_t == \widehat{q}_{t-1}$ then
3:     $h_t = h_{t-1}$
4: else
5:     Update the hypothesis by calling a weighted-ERM oracle:
           $h_t = \arg\min_{h \in \mathcal{H}} \sum_{i=1}^{t-1} \sum_{j=1}^{N} \frac{\widehat{q}_t(y_{i,j})}{\widehat{q}_i(y_{i,j})} \ell(h(x_{i,j}), y_{i,j})$    (B.106)
6: end if
7: Get $N$ co-variates $x_{t,1:N}$ and make predictions according to $h_t$
8: Get labels $y_{t,1:N}$
9: Compute $s_t[i] = \frac{1}{N} \sum_{j=1}^{N} \mathbb{I}\{y_{t,j} = i\}$ for all $i \in [K]$.
10: Update ALG with the empirical label marginals $s_t$.
```

Theorem B.5.3. *Assume the same notations as in Theorem 3.4.1. Suppose we run Algorithm 13 (see Appendix B.5) with ALG instantiated using Algorithm 12 with $\sigma^2 = 1/N$ and predictions clipped as in Theorem 3.4.1. Further let the online regression oracle given to Algorithm 12 be chosen as one of the candidates mentioned in Remark B.4.1. Then with probability at-least $1 - \delta$, we have that*

$$R^{\mathcal{H}}_{dynamic} = \widetilde{O}\left(T^{2/3} V_T^{1/3} + \sqrt{T \log(|\mathcal{H}|/\delta)}\right). \tag{B.107}$$

Further, the number of number of calls to ERM oracle (via Step 5) is at-most $\widetilde{O}(T^{1/3} V_T^{2/3})$ with probability at-least $1 - \delta$.

Sketch. The proof of this theorem closely follows the steps fused for proving Theorem 3.4.1. So we only highlight the changes that need to be incorporated to the proof of Theorem 3.4.1.

Replace the use of Proposition B.5.1 in the proof of Theorem 3.4.1 with Theorem B.4.1.

For any round t, where Step 5 of Algorithm 13 is triggered, we can use the same arguments as in the Proof of Theorem B.5.3 to bound the instantaneous regret by Eq.(B.83). i.e:

$$L_t(h_t) - L_t(h_t^*) \leqslant \frac{2B\sqrt{K}}{\mu^2} \left(\|\widehat{q}_t - q_t\|_2 + E[\|\widehat{q}_t - q_t\|_2] + \phi \cdot \frac{V_T^{1/3}}{(t-1)^{1/3}} + \sqrt{\frac{\log(3T|\mathcal{H}|/\delta)}{(t-1)}} \right), \tag{B.108}$$

with probability at-least $1 - \delta/T$.

For a round t, where Step 5 is not triggered, we proceed as follows:

Let t' be the most recent time step prior to t when Step 5 is executed. Notice that the population loss can be equivalently represented as

$$L_t(h) = \frac{1}{N(t'-1)} \sum_{i=1}^{t'-1} \sum_{j=1}^{N} E\left[\frac{q_t(y_{ij})}{q_i(y_{ij})} \ell(h(x_{ij}), y_{ij})\right], \tag{B.109}$$

where the expectation is taken with respect to randomness in the samples.

We define the following quantities:

$$L_t^{\text{emp}}(h) := \frac{1}{N(t'-1)} \sum_{i=1}^{t'-1} \sum_{j=1}^{N} \frac{q_t(y_{ij})}{q_i(y_{ij})} \ell(h(x_{ij}), y_{ij}), \tag{B.110}$$

$$\widetilde{L}_t(h) := \frac{1}{N(t'-1)} \sum_{i=1}^{t'-1} \sum_{j=1}^{N} E\left[\frac{\widehat{q}_t(y_{ij})}{\widehat{q}_i(y_{ij})} \ell(h(x_{ij}), y_{ij})\right], \tag{B.111}$$

and

$$\widetilde{L}_t^{\text{emp}}(h) := \frac{1}{N(t'-1)} \sum_{i=1}^{t'-1} \sum_{j=1}^{N} \frac{\widehat{q}_t(y_{ij})}{\widehat{q}_i(y_{ij})} \ell(h(x_{ij}), y_{ij}). \tag{B.112}$$

We decompose the regret at round t as

$$L_t(h_t) - L_t(h_t^*) = L_t(h_t) - \widetilde{L}_t(h_t) + \widetilde{L}_t(h_t) - \widetilde{L}_t^{\text{emp}}(h_t) + L_t^{\text{emp}}(h_t^*) - L_t(h_t^*) + \widetilde{L}_t^{\text{emp}}(h_t) - L_t^{\text{emp}}(h_t^*) \tag{B.113}$$

$$\leqslant \underbrace{L_t(h_t) - \widetilde{L}_t(h_t)}_{\text{T1}} + \underbrace{\widetilde{L}_t(h_t) - \widetilde{L}_t^{\text{emp}}(h_t)}_{\text{T2}} + \underbrace{L_t^{\text{emp}}(h_t^*) - L_t(h_t^*)}_{\text{T3}} + \underbrace{\widetilde{L}_t^{\text{emp}}(h_t^*) - L_t^{\text{emp}}(h_t^*)}_{\text{T4}}, \tag{B.114}$$

where in the last line we used Eq.(8.1). Now we proceed to bound each terms as note above.

By using the same arguments as in Proof of Theorem 3.4.1 and replacing the use of Proposition B.5.1 with Theorem B.4.1, we can bound T1-4. This will result in an instantaneous regret bound at round t (which doesn't trigger step 5) as:

$$L_t(h_t) - L_t(h_t^*) \leqslant \frac{2B\sqrt{K}}{\mu^2} \left(\|\widehat{q}_t - q_t\|_2 + E[|\widehat{q}_t - q_t\|_2] + \phi \cdot \frac{V_T^{1/3}}{(t'-1)^{1/3}} + \sqrt{\frac{\log(3T|\mathcal{H}|/\delta)}{(t'-1)}} \right), \tag{B.115}$$

$$\leqslant \frac{2B\sqrt{K}}{\mu^2}\left(\|\widehat{q}_t - q_t\|_2 + E[|\widehat{q}_t - q_t\|_2] + \phi \cdot 4^{1/3} \cdot \frac{V_T^{1/3}}{(t-1)^{1/3}} + \sqrt{\frac{4\log(3T|\mathcal{H}|/\delta)}{(t-1)}}\right), \tag{B.116}$$

with probability at-least $1 - \delta/T$. In the last line we used the fact that $t' - 1 \geqslant (t/2) - 1 \geqslant (t-1)/4$ for all $t \geqslant 3$.

Now adding Eq.(B.108) and (B.116) across all rounds and proceeding similar to the proof of Theorem 3.4.1 (and replacing the use of Proposition B.5.1 with Theorem B.4.1) completes the argument.

□

We next prove the matching (up to factors of $\log T$) lower bound.

Theorem 3.4.2. *Let $V_T \leqslant T/8$. There exists a choice of hypothesis class, loss function, and adversarial strategy of generating the data such that $R^{\mathcal{H}}_{dynamic} = \Omega\left(T^{2/3}V_T^{1/3} + \sqrt{T\log(|\mathcal{H}|)}\right)$, where the expectation is taken with respect to randomness in the algorithm and adversary.*

Proof. First we fix the hypothesis class and the data generation strategy. In the problem instance we consider, there are no co-variates. The hypothesis class is defined as

$$\mathcal{H} := \{h_p : h_p \text{ predicts a label } y \sim \text{Ber}(p);\ p \in [|\mathcal{H}|]\}. \tag{B.117}$$

Further we design the hypothesis class such that both $h_0, h_1 \in \mathcal{H}$. Next we fix the data generation strategy:

- Divide the time horizon into batches of length Δ.
- At the beginning of a batch i, the adversary picks $\mathring{q}_i$ uniformly at random from $\{1/2 - \delta, 1/2 + \delta\}$.
- For all rounds t that falls within batch i, the label $y_t \sim \text{Ber}(q_t)$ is sampled with $q_t := \mathring{q}_i$.
- Learner predicts a label $\widehat{y}_t \in \{0, 1\}$ and then the actual label y_t is revealed (hence $N = 1$ in the protocol of Fig.3).
- Learner suffers a loss given by $\ell_t(\widehat{y}_t) = \mathbb{I}\{\widehat{y}_t \neq y_t\}$.

It is easy to see that the losses are bounded in $[0, 1]$. Now let's examine the two possibilities of generating labels within a batch. Let's upper bound the variation incurred by the label marginals:

$$\sum_{t=2}^{T} |q_t - q_{t-1}| = \sum_{i=2}^{M} |\mathring{q}_i - \mathring{q}_{i-1}| \tag{B.118}$$

$$\leqslant 2\delta M \tag{B.119}$$

$$\leqslant V_T, \tag{B.120}$$

where the last line is obtained by choosing $\delta = V_T/(2M) = V_T\Delta/(2T)$.

Since at the beginning of each batch, the sampling probability of true labels is independently renewed, the historical data till the beginning of a batch is immaterial in minimising the loss within the batch. So we can lower bound the regret within each batch separately and add them up. Below, we focus on lower bounding the regret in batch 1 and the computations are similar for any other batch.

Suppose that the probability that an algorithm predict label $y_t = 1$ is $\widehat{q}_t$, where $\widehat{q}_t$ is a measurable function of the past data $y_{1:t-1}$. Then we have that the population loss $L_t(\widehat{q}_t) := E[\ell_t(\widehat{y}_t)] = (1-\widehat{q}_t)q_t + \widehat{q}_t(1-q_t)$. Here we abuse the notation $L(q_t) := L(h_{q_t})$. We see that the population loss $L_t(\widehat{q}_t)$ are convex and its gradient obeys $\nabla L_t(\widehat{q}_t) = 1 - 2q_t = E[1-2y_t]$ since by our construction $y_t \sim \text{Ber}(q_t)$. Thus the population losses are convex and its gradients can be estimated in an unbiased manner from the data.

We use the following Proposition due to Besbes et al. (2015).

Proposition B.5.4 (due to Lemma A-1 in Besbes et al. (2015))**.** *Let $\widetilde{\mathbb{P}}$ denote the joint probability of the label sequence $y_{1:\Delta}$ within a batch when they are generated using Ber$(1/2-\delta)$. So*

$$\widetilde{\mathbb{P}}(y_1, \ldots, y_\Delta) = \Pi_{i=1}^{\Delta}(1/2-\delta)^{y_i}(1/2+\delta)^{1-y_i}. \tag{B.121}$$

Similarly define $\widetilde{\mathbb{Q}}$ as

$$\widetilde{\mathbb{Q}}(y_1, \ldots, y_\Delta) = \Pi_{i=1}^{\Delta}(1/2+\delta)^{y_i}(1/2-\delta)^{1-y_i}. \tag{B.122}$$

According to the above distribution the data are independently sampled from Bernoulli trials with success probability $1/2+\delta$. Let $\widehat{q}_t$ be the decision of the online algorithm qt round t so that the algorithm predicts label 1 with probability $\widehat{q}_t$.

Let $\mathbb{P}$ denote the joint probability distribution across the decisions $\widehat{q}_{1:\Delta}$ of any online algorithm under the sampling model $\widetilde{\mathbb{P}}$. Similarly define $\mathbb{Q}$. Note that any online algorithm can make decisions at round t only based on the past observed data $y_{1:t-1}$. Further after making the decision $\widehat{q}_t$ at round t, an unbiased estimate of the population loss can be constructed due to the fact that $\nabla L_t(\widehat{q}_t) = E[1-2y_t]$. Under the availability of unbiased gradient estimates of the losses, it holds that

$$KL(\mathbb{P}||\mathbb{Q}) \leqslant 4\Delta\delta^2. \tag{B.123}$$

By choosing $\delta = V_T/(2M) = V_T\Delta/(2T)$ and $\Delta = (T/V_T)^{2/3}$, we get that $KL(\mathbb{P}||\mathbb{Q}) \leqslant 1$.

Since $V_T \leqslant T/8$, the above choice implies that $\delta \in (0, 1/4)$.

For notational clarity, define $L^{\mathbb{P}}(q) = (1-q)(1/2-\delta) + q(1/2+\delta)$ and $L^{\mathbb{Q}}(q) = (1-q)(1/2+\delta) + q(1/2-\delta)$. These corresponds to the population losses according to the sampling models $\mathbb{P}$ and $\mathbb{Q}$ respectively. Observe that $\min_q L^{\mathbb{P}}(q) = \min_q L^{\mathbb{Q}}(q) = 1/2 - \delta$. The minimum of $L^{\mathbb{P}}$ and $L^{\mathbb{Q}}$ are achieved at 0 and 1 respectively. Note that both $h_0, h_1 \in \mathcal{H}$. So there is always a hypothesis in $\mathcal{H}$ that corresponds the minimiser of the loss.

Further whenever $\widehat{q} \geqslant 1/2$ we have that

$$\begin{aligned} L^{\mathbb{P}}(q) &= (1/2 - \delta) + q(2\delta) && \text{(B.124)} \\ &\geqslant 1/2. && \text{(B.125)} \end{aligned}$$

Similarly whenever $q < 1/2$ we have $L^{\mathbb{Q}}(q) \geqslant 1/2$. So we define the selector function as $\phi_t := \mathbb{I}\{\widehat{q}_t \geqslant 1/2\}$. Let $q_t^* \in \{0, 1\}$ be the minimiser of the loss at round t. Now we can lower bound the instantaneous regret similar as

$$\begin{aligned} E[L_t(\widehat{q}_t) - L_t(q_t^*)] &= \frac{1}{2}(E_{\mathbb{P}}[L_t^{\mathbb{P}}(\widehat{q}_t) - L_t^{\mathbb{P}}(0)] + \frac{1}{2}(E_{\mathbb{Q}}[L_t^{\mathbb{Q}}(\widehat{q}_t) - L_t^{\mathbb{Q}}(1)] && \text{(B.126)} \\ &\geqslant \frac{1}{2}(E_{\mathbb{P}}[L_t^{\mathbb{P}}(\widehat{q}_t) - L_t^{\mathbb{P}}(0)|\phi_t = 1]\mathbb{P}(\phi_t = 1) + \frac{1}{2}(E_{\mathbb{Q}}[L_t^{\mathbb{Q}}(\widehat{q}_t) - L_t^{\mathbb{Q}}(1)|\phi_t = 0]\mathbb{Q}(\phi_t = 0) && \text{(B.127)} \\ &\geqslant \delta/2 \max\{\mathbb{P}(\phi_t = 1), \mathbb{Q}(\phi_t = 0)\} && \text{(B.128)} \\ &\geqslant (\delta/8)e^{-1}, && \text{(B.129)} \end{aligned}$$

where the last line is obtained by Propositions B.5.4 and B.3.5.

Thus we get a total lower bound on the instantanoeus regret as

$$\begin{aligned} \sum_{t=1}^{T} E[L_t(\widehat{q}_t) - L_t(q_t^*)] &\geqslant T\delta/(8e) && \text{(B.130)} \\ &= \Delta V_T/(16e) && \text{(B.131)} \\ &= T^{2/3} V_T^{1/3}/(16e), && \text{(B.132)} \end{aligned}$$

where the last line is obtained by using our choices of $\delta V_T \Delta/(2T)$ and $\Delta = (T/V_T)^{2/3}$.

The second term of of $\Omega(\sqrt{T \log |\mathcal{H}|})$ can be obtained from the existing results on statistical learning theory without distribution shifts. (see for example Theorem 3.23 in Mohri et al. (2012)).

□

B.6 More details on experiments

In Algorithm Algorithm 14, we describe the FLH-FTL algorithm from Hazan and Seshadhri (2007) when specialised to squared error losses. When specialized to squared error losses, this algorithm runs FLH with online averages as the base experts.

Algorithm 14 An instance of FLH-FTL from Hazan and Seshadhri (2007) with squared error losses

```
Parameter $\alpha$ is defined to be a learning rate
// initializations and definitions
For FLH-FTL instantiations within UOLS algorithms (as in Algorithm 2), we set $\alpha \leftarrow \sigma_{min}^2(C)/(8K)$, where $C$ is the confusion matrix as in Assumption 1. For instantiations within SOLS algorithms (as in Algorithm 4) we set $\alpha \leftarrow 1/(8K)$
For each round $t \in [T]$, $v_t := (v_t^{(1)}, \ldots, v_t^{(t)})$ is a probability vector in $\mathbb{R}^t$. Initialize $v_1^{(1)} \leftarrow 1$
For each $j \in [T]$, define a base learner $E^j$. For each $t > j$, the base expert outputs $E^j(t) := \frac{1}{t-j} \sum_{i=j}^{t-1} z_j$, where $z_j$ to be specified as below. Further $E^j(j) := 0 \in \mathbb{R}^K$
// execution steps
In round $t \in [T]$, set $\forall j \leqslant t$, $x_t^j \leftarrow E^j(t)$ (the prediction of the $j^{th}$ base learner at time $t$). Play $x_t = \sum_{j=1}^t v_t^{(j)} x_t^{(j)}$.
Receive feedback $z_t$, set $\widehat{v}_{t+1}^{(t+1)} \leftarrow 0$ and perform update for $1 \leqslant i \leqslant t$:
    $\widehat{v}_{t+1}^{(i)} \leftarrow \frac{v_t^{(i)} e^{-\alpha \|x_t^{(i)} - z_t\|_2^2}}{\sum_{j=1}^t v_t^{(j)} e^{-\alpha \|x_t^{(j)} - z_t\|_2^2}}$    (B.133)
Addition step - Set $v_{t+1}^{(t+1)}$ to $1/(t+1)$ and for $i \neq t+1$:
    $v_{t+1}^{(i)} \leftarrow (1 - (t+1)^{-1}) \widehat{v}_{t+1}^{(i)}$    (B.134)
```

Rationale behind the learning rate setting at Line 2 of Algorithm 14 The loss that is incurred by Algorithm 14 and any of its base learners at round t is defined to be the squared error loss $\ell_t(x) = \|z_t - x\|_2^2$. Whenever $\|z_t\|_2^2 \leqslant B^2$ and $\|x\|_2^2 \leqslant B^2$, the losses $\ell_t(x)$ are $1/(8B^2)$ exp-concave (see for eg. Chapter 3 of (Cesa-Bianchi and Lugosi, 2006)). The notion of exp-concavity is crucial for FLH-FTL algorithm since the learning rate is set to be equal to the exp-concavity factor of the loss functions (see Theorem 3.1 in Hazan and Seshadhri (2007)).

For the UOLS problem, from Algorithm 2, we have $\|z_t\|_2 = \|C^{-1} f_0(x_t)\|_2 \leqslant \sqrt{K}/\sigma_{min}(C)$. Since the decisions of the algorithm is a convex combination of the previously seen z_t, we conclude that the losses $\ell_t(x)$ are $\sigma_{min}^2(C)/(8K)$ exp-concave.

For the SOLS problem, let $z_t = s_t$ where s_t is as defined in Algorithm 4. We have that $\|z_t\|_2 \leqslant \sqrt{K}$. Hence arguing in a similar fashion as above, we conclude that the losses $\ell_t(x)$ are $1/(8K)$ exp-concave for the SOLS problem.

This is the motivation behind Line 2 in Algorithm 14, where the learning rates are set according to the problem setting.

Dataset and model details.

- Synthetic: For the synthetic data, we generated 72k samples as described in Bai et al. (2022). There are three classes each with 24k samples generated from three Gaussian distributions in $\mathbb{R}^{12}$. Each Gaussian distribution is defined by a randomly generated unit-norm centre v and covariance matrix $0.215 \cdot I$. 60k samples are used as source data, and 12k samples are used as target data to be sampled from during online learning. We used logistic regression to train a linear model. It is trained for a single epoch with learning rate 0.1, momentum 0.9, batch size 200, and l_2 regularization 1×10^{-4}.

- MNIST (LeCun et al., 1998): An image dataset of 10 types of handwritten digits. 60k samples are used as source data and 10k as target data. We used an MLP for prediction with three consecutive hidden layers of sizes 100, 100, and 20. It is trained for a single epoch with a learning rate 0.1, momentum 0.9, batch size 200, and l_2 regularization 1×10^{-4}.

- CIFAR-10 (Krizhevsky and Hinton, 2009): A dataset of colored images of 10 items: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. 50k samples are used as source data and 10k as target data. We train a ResNet18 model (He et al. (2016)) from scratch. It is finetuned for 70 epochs with learning rate 0.1, momentum 0.9, batch size 200, and l_2 regularization 1×10^{-4}. The learning rate decayed by 90% at the 25th and 40th epochs.

- Fashion (Xiao et al., 2017): An image dataset of 10 types of fashion items: T-shirt, trouser, pullover, dress, coat, sandals, shirt, sneaker, bag, and ankle boots. 60k samples are used as source data and 10k as target data. We trained an MLP for prediction. It is trained for 50 epochs with learning rate 0.1, momentum 0.9, batch size 200, and l_2 regularization 1×10^{-4}.

- EuroSAT (Helber et al., 2019): An image dataset of 10 types of land uses: industrial buildings, residential buildings, annual crop, permanent crop, river, sea & lake, herbaceous vegetation, highway, pasture, and forest. 60k samples are used as source data and 10k as target data. We cropped the images to the size $(3, 64, 64)$. We train a ResNet18 model for 50 epochs with learning rate 0.1, momentum 0.9, batch size 200, and l_2 regularization 1×10^{-4}.
- Arxiv (Clement et al., 2019): A natural language dataset of 23 classes over different publication subjects. 198k samples are used as source data and 22k as target data. We trained a DistilBERT model (Sanh et al. (2019b)) for 50 epochs with learning rate 2×10^{-5}, batch size 64, and l_2 regularization 1×10^{-2}.
- SHL (Gjoreski et al., 2018; Wang et al., 2019c): A tabular locomotion dataset of 6 classes of human motion: still, walking, run, bike, car and bus. 30k samples are used as source data and 70k as target data. We trained an MLP for prediction for 50 epochs with learning rate 0.1, momentum 0.9, batch size 200, and l_2 regularization 1×10^{-4}.

For all the datasets above, the initial offline data are further split by $80 : 20$ into training and holdout data, where the former is used for offline training of the base model and the latter for computing the confusion matrix and retraining (e.g. updating the linear head parameters with UOGD or updating the softmax prediction with our FLT-FTL) during online learning. To examine how well the algorithms adapt when holdout data is limited, we use 10% of the holdout data (i.e., 2% of the initial offline data) in the main paper unless stated otherwise. In App. **??**, we ablate with full hold-out data.

Types of Simulated Shifts. We simulate four kinds of label shifts to capture different non-stationary environments. These shifts are similar to the ones used in Bai et al. (2022). For each shift, the label marginals are a mixture of two different constant marginals $\mu_1, \mu_2 \in \Delta_K$ with a time-varying coefficient α_t: $\mu_{y_t} = (1 - \alpha_t)\mu_1 + \alpha_t \mu_2$, where μ_{y_t} denotes the label distribution at round t and α_t controls the shift non-stationarity and patterns. In particular, we have: *Sinusoidal Shift (Sin)*: $\alpha_t = \sin \frac{i\pi}{L}$, where $i = t \mod L$ and L is a given periodic length. In the experiments, we set $L = \sqrt{T}$. *Bernoulli Shift (Ber)*: at every iteration, we keep the $\alpha_t = \alpha_{t-1}$ with probability $p \in [0, 1]$ and otherwise set $\alpha_t = 1 - \alpha_{t-1}$. In the experiments, the parameter is set as $p = 1/\sqrt{T}$, which implies $V_t = \Theta(\sqrt{T})$. *Square Shift (Squ)*: at every L rounds we set $\alpha_t = 1 - \alpha_{t-1}$. *Monotone Shift (Mon)*: we set $\alpha_t = t/T$. Square, sinusoidal, and Bernoulli shifts simulate fast-changing environments with periodic patterns.

Methods for UOLS Adaptation.

- Base: the base classifier without any online modifications.

		CT	CT-RS (ours)	CT-RS (ours)	w-ERM
		(base)	w FLH	w FLT-FTL	(oracle)
MNIST	Cl Err	5.0±0.5	4.71±0.2	**4.53±0.1**	3.2±0.4
	MSE	NA	0.12±0.01	**0.08±0.01**	NA

Table B.2: *Results on SOLS setup.* We report results with MNIST SOLS setup runs for $T = 200$ steps. We observe that continual training with re-sampling improves over the base model which continually trains on the online data and achieves competitive performance with respect to weighted ERM oracle.

- OFC: the optimal fixed classifier predicts with an optimal fixed re-weighting in hindsight as in Wu et al. (2021).
- Oracle: re-weight the base model's predictions with the true label marginal of the unlabeled data at each time step.
- FTH: proposed by Wu et al. (2021), follow-the-history classifier re-weights the base model's predictions with a simple online average of all marginal estimates seen thus far.
- FTFWH: proposed by Wu et al. (2021), follow-the-fixed-window-history classifier is a version of FTH that tracks only the k most recent marginal estimates. We choose $k = 100$ throughout the experiments in this work.
- ROGD: proposed by Wu et al. (2021), ROGD uses online gradient descent to update its re-weighting vector based on current marginal estimate.
- UOGD: proposed by Bai et al. (2022), retrains the last linear layer of the model based on current marginal estimate.
- ATLAS: proposed by Bai et al. (2022) is a meta-learning algorithm that has UOGD as its base learners.

The learning rates of ROGD, UOGD, and ATLAS are set according to suggestions in the original works. The learning rate of FLH-FTL is set to $1/K$. This corresponds to a faster rate than the theoretically optimal learning rate given in Line 2 of Algorithm 14. It has been observed in prior works such as Baby et al. (2021) that the theoretical learning rate is often too conservative and faster rates lead to a better performance.

B.6.1 Supervised Online Label Shift Experiment Details

For each dataset, we first fix the number of time steps and then simulate the label marginal shift. To train the learner with all the methods, we store all the online data observed giving the storage complexity of $\mathcal{O}(T)$. We observe $N = 50$ examples at every iteration and we split the observed labeled examples into 80:20 split for training and validation. The validation examples are used to decide the number of gradient steps at every time step, in

	CT-RS (ours)	w-ERM (oracle)
CIFAR	$\mathbf{145}_{\pm \mathbf{3.7}}$	$1882_{\pm 14}$
MNIST	$\mathbf{20}_{\pm \mathbf{2.7}}$	$107_{\pm 3.6}$

Table B.3: *Comparison on computation time (in minutes).* We report results with MNIST and CIFAR SOLS setup runs for $T = 200$ steps. We observe that continual training with re-sampling is approximately 5–15× more efficient than weighted ERM oracle.

particular, we take gradient steps till the validation error continues to decrease.

Dataset and model details.

- MNIST (LeCun et al., 1998): An image dataset of 10 types of handwritten digits. At each step, we sample 50 samples with the label marginal that step without replacement and reveal the examples to the learner. We used an MLP for prediction with three consecutive hidden layers of sizes 100, 100, and 20. It is trained for a single epoch with a learning rate 0.1, momentum 0.9, batch size 200, and l_2 regularization 1×10^{-4}.
- CIFAR-10 (Krizhevsky and Hinton, 2009): A dataset of colored images of 10 items: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. At each step, we sample 50 samples with the label marginal that step without replacement and reveal the examples to the learner. It is finetuned for 70 epochs with learning rate 0.1, momentum 0.9, batch size 200, and l_2 regularization 1×10^{-4}.

We simulate Bernoulli label shifts to capture different non-stationary environments.

Connection of CT-RS to weighted ERM Before making the connection, we first re-visit the CT-RS algorithm. **Step 1**: Maintain a pool of all the labeled data received till that time step, and at every iteration, we randomly sample a batch with uniform label marginal to update the model. **Step 2**: Re-weight the softmax outputs of the updated model with estimated label marginal. Below we show that it is equivalent to wERM:

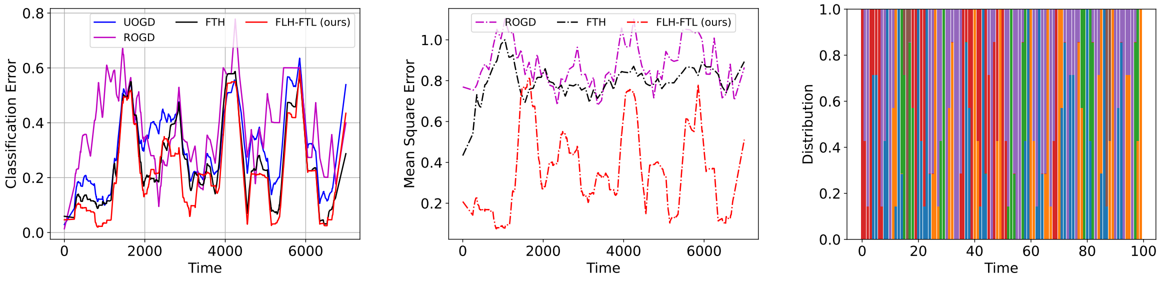
$$\begin{aligned}
f_t &= \arg\min_{f \in \mathcal{H}} \sum_{i=1}^{t-1} \sum_{j=1}^{N} \frac{\widehat{q}_t(y_{i,j})}{\widehat{q}_i(y_{i,j})} \ell(f(x_{i,j}), y_{i,j}) \\
&= \arg\min_{f \in \mathcal{H}} \sum_{k=1}^{K} \widehat{q}_t(k) \sum_{i=1}^{t-1} \sum_{j=1}^{N} \frac{\mathbb{1}\left(y_{i,j} = k\right)}{\widehat{q}_i(k)} \ell(f(x_{i,j}), k) \\
&= \arg\min_{f \in \mathcal{H}} \sum_{k=1}^{K} \frac{\widehat{q}_t(k)}{(1/K)} \sum_{i=1}^{t-1} \sum_{j=1}^{N} \frac{\mathbb{1}\left(y_{i,j} = k\right)}{K \cdot \widehat{q}_i(k)} \ell(f(x_{i,j}), k)
\end{aligned}$$

$$= \arg\min_{f\in\mathcal{H}} \sum_{k=1}^{K} \widehat{\mu}_{t,k} \underbrace{\sum_{i=1}^{t-1}\sum_{j=1}^{N} \frac{\mathbb{1}\left(y_{i,j}=k\right)}{\widehat{\mu}_{i,k}} \ell(f(x_{i,j}),k)}_{L_{t-1,k}}$$

where $\widehat{\mu}_{t,k} = {}^{\widehat{q}_t(k)}/_{(1/K)}$ is the importance ratio at time i with respect to uniform label marginal. Similarly, we define $\widehat{\mu}_{i,k} = {}^{\widehat{q}_i(k)}/_{(1/K)}$. Here, $L_{t-1,k}$ is the aggregate loss at t-th time step for k-th class such that at each step the sampling probability of that class is uniform. By continually training a classifier with CT-RS, Step 1 approximates the classifier $\widetilde{f}_t$ trained to minimize the average of $L_{t-1,k}$ over all classes with uniform proportion for each class. To update the classifier $\widetilde{f}_t$ according to label proportions at time t, we update the softmax output of $\widetilde{f}_t$ according to $\widehat{\mu}_t$.

The primary benefit of CT-RS over wERM is to avoid re-training from scratch at every iteration. Instead, we can leverage the model trained in the previous iteration to warm-start training in the next iteration.

B.6.2 Additional results and details on the SHL dataset



(a) (b) (c)

Figure B.1: *Additional results and details on the SHL datasets with real shift.* **(a) and (b):** The accuracies and mean square errors in label marginal estimation on SHL dataset over 7,000 time steps with limited amount of holdout data. **(c):** Label marginals of the six classes of SHL dataset. Each time step here shows the marginals over 700 samples.

Appendix C

Appendix: Mixture Proportion Estimation and PU Learning: A Modern Approach

C.1 Proofs from Sec. 4.4

Proof of Lemma D.4.2. The proof primarily involves using DKW inequality (Dvoretzky et al., 1956) on $\widehat{w}q_u(c)$ and $\widehat{w}q_p(c)$ to show convergence to their respective means $q_u(c)$ and $q_p(c)$. First, we have

$$\begin{aligned}\left|\frac{\widehat{w}q_u(c)}{\widehat{w}q_p(c)} - \frac{q_u(c)}{q_p(c)}\right| &= \frac{1}{\widehat{w}q_u(c)\cdot q_u(c)}\left|\widehat{w}q_u(c)\cdot q_p(c) - q_p(c)\cdot q_u(c) + q_p(c)\cdot q_u(c) - \widehat{w}q_p(c)\cdot q_u(c)\right| \\ &\leqslant \frac{1}{\widehat{w}q_p(c)}\left|\widehat{w}q_u(c) - q_u(c)\right| + \frac{q_u(c)}{\widehat{w}q_p(c)\cdot q_u(c)}\left|\widehat{w}q_p(c) - q_p(c)\right| . \qquad \text{(C.1)}\end{aligned}$$

Using DKW inequality, we have with probability $1-\delta$: $|\widehat{w}q_p(c) - q_p(c)| \leqslant \sqrt{\frac{\log(2/\delta)}{2n_p}}$ for all $c \in [0,1]$. Similarly, we have with probability $1-\delta$: $|\widehat{w}q_u(c) - q_u(c)| \leqslant \sqrt{\frac{\log(2/\delta)}{2n_u}}$ for all $c \in [0,1]$. Plugging this in (C.1), we have

$$\left|\frac{\widehat{w}q_u(c)}{\widehat{w}q_p(c)} - \frac{q_u(c)}{q_p(c)}\right| \leqslant \frac{1}{\widehat{w}q_p(c)}\left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + \frac{q_u(c)}{q_p(c)}\sqrt{\frac{\log(4/\delta)}{2n_p}}\right) .$$

□

Proof of Theorem 4.4.1. The main idea of the proof is to use the confidence bound derived in Lemma D.4.2 at $\widehat{w}c$ and use the fact that $\widehat{w}c$ minimizes the upper confidence bound. The proof is split into two parts. First, we derive a lower bound on $\widehat{w}q_p(\widehat{w}c)$ and next, we use the obtained lower bound to derive confidence bound on $\widehat{w}\alpha$. All the statements in the proof simultaneously hold with probability $1-\delta$. Recall,

$$\widehat{w}c := \arg\min_{c\in[0,1]} \frac{\widehat{w}q_u(c)}{\widehat{w}q_p(c)} + \frac{1}{\widehat{w}q_p(c)}\left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + (1+\gamma)\sqrt{\frac{\log(4/\delta)}{2n_p}}\right) \qquad \text{and} \tag{C.2}$$

$$\widehat{w}\alpha := \frac{\widehat{w}q_u(\widehat{w}c)}{\widehat{w}q_p(\widehat{w}c)}\,. \tag{C.3}$$

Moreover,

$$c^* := \arg\min_{c\in[0,1]} \frac{q_u(c)}{q_p(c)} \qquad \text{and} \qquad \alpha^* := \frac{q_u(c^*)}{q_p(c^*)}\,. \tag{C.4}$$

Part 1: We establish lower bound on $\widehat{w}q_p(\widehat{w}c)$. Consider $c' \in [0,1]$ such that $\widehat{w}q_p(c') = \frac{\gamma}{2+\gamma}\widehat{w}q_p(c^*)$. We will now show that Algorithm 17 will select $\widehat{w}c < c'$. For any $c \in [0,1]$, we have with with probability $1-\delta$,

$$\widehat{w}q_p(c) - \sqrt{\frac{\log(4/\delta)}{2n_p}} \leqslant q_p(c) \qquad \text{and} \qquad q_u(c) - \sqrt{\frac{\log(4/\delta)}{2n_u}} \leqslant \widehat{w}q_u(c)\,. \tag{C.5}$$

Since $\frac{q_u(c^*)}{q_p(c^*)} \leqslant \frac{q_u(c)}{q_p(c)}$, we have

$$\widehat{w}q_u(c) \geqslant q_p(c)\frac{q_u(c^*)}{q_p(c^*)} - \sqrt{\frac{\log(4/\delta)}{2n_u}} \geqslant \left(\widehat{w}q_p(c) - \sqrt{\frac{\log(4/\delta)}{2n_p}}\right)\frac{q_u(c^*)}{q_p(c^*)} - \sqrt{\frac{\log(4/\delta)}{2n_u}}\,. \tag{C.6}$$

Therefore, at c we have

$$\frac{\widehat{w}q_u(c)}{\widehat{w}q_p(c)} \geqslant \alpha^* - \frac{1}{\widehat{w}q_p(c)}\left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + \frac{q_u(c^*)}{q_p(c^*)}\sqrt{\frac{\log(4/\delta)}{2n_p}}\right)\,. \tag{C.7}$$

Using Lemma D.4.2 at c^*, we have

$$\frac{\widehat{w}q_u(c)}{\widehat{w}q_p(c)} \geqslant \frac{\widehat{w}q_u(c^*)}{\widehat{w}q_p(c^*)} - \left(\frac{1}{\widehat{w}q_p(c^*)} + \frac{1}{\widehat{w}q_p(c)}\right)\left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + \frac{q_u(c^*)}{q_p(c^*)}\sqrt{\frac{\log(4/\delta)}{2n_p}}\right) \tag{C.8}$$

$$\geqslant \frac{\widehat{w}q_u(c^*)}{\widehat{w}q_p(c^*)} - \left(\frac{1}{\widehat{w}q_p(c^*)} + \frac{1}{\widehat{w}q_p(c)}\right)\left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + \sqrt{\frac{\log(4/\delta)}{2n_p}}\right)\,, \tag{C.9}$$

where the last inequality follows from the fact that $\alpha^* = \frac{q_u(c^*)}{q_p(c^*)} \leqslant 1$. Furthermore, the upper confidence bound at c is lower bound as follows:

$$\frac{\widehat{w}q_u(c)}{\widehat{w}q_p(c)} + \frac{1+\gamma}{\widehat{w}q_p(c)}\left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + \sqrt{\frac{\log(4/\delta)}{2n_p}}\right) \tag{C.10}$$

$$\geqslant \frac{\widehat{w}q_u(c^*)}{\widehat{w}q_p(c^*)} + \left(\frac{1+\gamma}{\widehat{w}q_p(c)} - \frac{1}{\widehat{w}q_p(c^*)} - \frac{1}{\widehat{w}q_p(c)} \right) \left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + \sqrt{\frac{\log(4/\delta)}{2n_p}} \right) \tag{C.11}$$

$$= \frac{\widehat{w}q_u(c^*)}{\widehat{w}q_p(c^*)} + \left(\frac{\gamma}{\widehat{w}q_p(c)} - \frac{1}{\widehat{w}q_p(c^*)} \right) \left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + \sqrt{\frac{\log(4/\delta)}{2n_p}} \right) \tag{C.12}$$

Using (D.13) at $c = c'$, we have the following lower bound on ucb at c':

$$\frac{\widehat{w}q_u(c')}{\widehat{w}q_p(c')} + \frac{1+\gamma}{\widehat{w}q_p(c')} \left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + \sqrt{\frac{\log(4/\delta)}{2n_p}} \right) \tag{C.13}$$

$$\geqslant \frac{\widehat{w}q_u(c^*)}{\widehat{w}q_p(c^*)} + \frac{1+\gamma}{\widehat{w}q_p(c^*)} \left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + \sqrt{\frac{\log(4/\delta)}{2n_p}} \right) , \tag{C.14}$$

Moreover from (D.13), we also have that the lower bound on ucb at $c \geqslant c'$ is strictly greater than the lower bound on ucb at c'. Using definition of $\widehat{w}c$, we have

$$\frac{\widehat{w}q_u(c^*)}{\widehat{w}q_p(c^*)} + \frac{1+\gamma}{\widehat{w}q_p(c^*)} \left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + \sqrt{\frac{\log(4/\delta)}{2n_p}} \right) \tag{C.15}$$

$$\geqslant \frac{\widehat{w}q_u(\widehat{w}c)}{\widehat{w}q_p(\widehat{w}c)} + \frac{1+\gamma}{\widehat{w}q_p(\widehat{w}c)} \left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + \sqrt{\frac{\log(4/\delta)}{2n_p}} \right) , \tag{C.16}$$

and hence

$$\widehat{w}c \leqslant c' \, . \tag{C.17}$$

Part 2: We now establish an upper and lower bound on $\widehat{w}\alpha$. We start with upper confidence bound on $\widehat{w}\alpha$. By definition of $\widehat{w}c$, we have

$$\frac{\widehat{w}q_u(\widehat{w}c)}{\widehat{w}q_p(\widehat{w}c)} + \frac{1+\gamma}{\widehat{w}q_p(\widehat{w}c)} \left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + \sqrt{\frac{\log(4/\delta)}{2n_p}} \right) \tag{C.18}$$

$$\leqslant \min_{c\in[0,1]} \left[\frac{\widehat{w}q_u(c)}{\widehat{w}q_p(c)} + \frac{1+\gamma}{\widehat{w}q_p(c)} \left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + \sqrt{\frac{\log(4/\delta)}{2n_p}} \right) \right] \tag{C.19}$$

$$\leqslant \frac{\widehat{w}q_u(c^*)}{\widehat{w}q_p(c^*)} + \frac{1+\gamma}{\widehat{w}q_p(c^*)} \left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + \sqrt{\frac{\log(4/\delta)}{2n_p}} \right) . \tag{C.20}$$

Using Lemma D.4.2 at c^*, we get

$$\frac{\widehat{w}q_u(c^*)}{\widehat{w}q_p(c^*)} \leqslant \frac{q_u(c^*)}{q_p(c^*)} + \frac{1}{\widehat{w}q_p(c^*)} \left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + \frac{q_u(c^*)}{q_p(c^*)} \sqrt{\frac{\log(4/\delta)}{2n_p}} \right)$$

$$= \alpha^* + \frac{1}{\widehat{w}q_p(c^*)} \left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + \alpha^* \sqrt{\frac{\log(4/\delta)}{2n_p}} \right) . \tag{C.21}$$

Combining (D.21) and (D.22), we get

$$\widehat{w}\alpha = \frac{\widehat{w}q_u(\widehat{w}c)}{\widehat{w}q_p(\widehat{w}c)} \leqslant \alpha^* + \frac{2+\gamma}{\widehat{w}q_p(c^*)} \left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + \sqrt{\frac{\log(4/\delta)}{2n_p}} \right) . \tag{C.22}$$

Using DKW inequality on $\widehat{w}q_p(c^*)$, we have $\widehat{w}q_p(c^*) \geqslant q_p(c^*) - \sqrt{\frac{\log(4/\delta)}{2n_p}}$. Assuming $n_p \geqslant \frac{2\log(4/\delta)}{q_p^2(c^*)}$, we get $\widehat{w}q_p(c^*) \leqslant q_p(c^*)/2$ and hence,

$$\widehat{w}\alpha \leqslant \alpha^* + \frac{4+2\gamma}{q_p(c^*)} \left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + \sqrt{\frac{\log(4/\delta)}{2n_p}} \right) . \tag{C.23}$$

Finally, we now derive a lower bound on $\widehat{w}\alpha$. From Lemma D.4.2, we have the following inequality at $\widehat{w}c$

$$\frac{q_u(\widehat{w}c)}{q_p(\widehat{w}c)} \leqslant \frac{\widehat{w}q_u(\widehat{w}c)}{\widehat{w}q_p(\widehat{w}c)} + \frac{1}{\widehat{w}q_p(\widehat{w}c)} \left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + \frac{q_u(\widehat{w}c)}{q_p(\widehat{w}c)} \sqrt{\frac{\log(4/\delta)}{2n_p}} \right) . \tag{C.24}$$

Since $\alpha^* \leqslant \frac{q_u(\widehat{w}c)}{q_p(\widehat{w}c)}$, we have

$$\alpha^* \leqslant \frac{q_u(\widehat{w}c)}{q_p(\widehat{w}c)} \leqslant \frac{\widehat{w}q_u(\widehat{w}c)}{\widehat{w}q_p(\widehat{w}c)} + \frac{1}{\widehat{w}q_p(\widehat{w}c)} \left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + \frac{q_u(\widehat{w}c)}{q_p(\widehat{w}c)} \sqrt{\frac{\log(4/\delta)}{2n_p}} \right) . \tag{C.25}$$

Using (D.24), we obtain a very loose upper bound on $\frac{\widehat{w}q_u(\widehat{w}c)}{\widehat{w}q_p(\widehat{w}c)}$. Assuming $\min(n_p, n_u) \geqslant \frac{2\log(4/\delta)}{q_p^2(c^*)}$, we have $\frac{\widehat{w}q_u(\widehat{w}c)}{\widehat{w}q_p(\widehat{w}c)} \leqslant \alpha^* + 4 + 2\gamma \leqslant 5 + 2\gamma$. Using this in (D.26), we have

$$\alpha^* \leqslant \frac{\widehat{w}q_u(\widehat{w}c)}{\widehat{w}q_p(\widehat{w}c)} + \frac{1}{\widehat{w}q_p(\widehat{w}c)} \left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + (5+2\gamma) \sqrt{\frac{\log(4/\delta)}{2n_p}} \right) . \tag{C.26}$$

Moreover, as $\widehat{w}c \geqslant c'$, we have $\widehat{w}q_p(\widehat{w}c) \geqslant \frac{\gamma}{2+\gamma} \widehat{w}q_p(c^*)$ and hence,

$$\alpha^* - \frac{\gamma+2}{\gamma \widehat{w}q_p(c^*)} \left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + (5+2\gamma) \sqrt{\frac{\log(4/\delta)}{2n_p}} \right) \leqslant \frac{\widehat{w}q_u(\widehat{w}c)}{\widehat{w}q_p(\widehat{w}c)} = \widehat{w}\alpha \, . \tag{C.27}$$

As we assume $n_p \geqslant \frac{2\log(4/\delta)}{q_p^2(c^*)}$, we have $\widehat{w}q_p(c^*) \leqslant q_p(c^*)/2$, which implies the following lower bound on α:

$$\alpha^* - \frac{2\gamma+4}{\gamma q_p(c^*)} \left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + (5+2\gamma) \sqrt{\frac{\log(4/\delta)}{2n_p}} \right) \leqslant \widehat{w}\alpha \, . \tag{C.28}$$

□

Proof of Corollary 4.4.3. Note that since $\alpha \leqslant \alpha^*$, the lower bound remains the same as in Theorem 4.4.1. For upper bound, plugging in $q_u(c) = \alpha q_p(c) + (1-\alpha) q_n(c)$, we have $\alpha^* = \alpha + (1-\alpha) q_n(c^*)/q_p(c^*)$ and hence, the required upper bound. □

C.1.1 Note on γ in Algorithm 17

We multiply the upper bound in Lemma D.4.2 to establish lower bound on $\widehat{w}q_p(\widehat{w}c)$. Otherwise, in an extreme case, with $\gamma = 0$, Algorithm 17 can select $\widehat{w}c$ with arbitrarily low $\widehat{w}q_p(\widehat{w}c)$ ($\ll q_p(c^*)$) and hence poor concentration guarantee to the true mixture proportion. However, with a small positive γ, we can obtain lower bound on $\widehat{w}q_p(\widehat{w}c)$ and hence tight guarantees on the ratio estimate ($\widehat{w}q_u(\widehat{w}c)/\widehat{w}q_p(\widehat{w}c)$) in Theorem 4.4.1.

In our experiments, we choose $\gamma = 0.01$. However, we didn't observe any (significant) differences in mixture proportion estimation even with $\gamma = 0$. implying that we never observe $\widehat{w}q_p(\widehat{w}c)$ taking arbitrarily small values in our experiments.

C.2 Comparison of BBE with Scott (2015)

Heuristic estimator due to Scott (2015) is motivated by the estimator in Blanchard et al. (2010). The estimator in Blanchard et al. (2010) relies on VC bounds, which are known to be loose in typical deep learning situations. Therefore, Scott (2015) proposed an heuristic implementation based on the minimum slope of any point in the ROC space to the point $(1, 1)$. To obtain ROC estimates, authors use direct binomial tail inversion (instead of VC bounds as in Blanchard et al. (2010)) to obtain tight upper bounds for true positives and lower bounds for true negatives. Finally, using these conservatives estimates the estimator in Scott (2015) is obtained as the minimum slope of any of the operating points to the point $(1, 1)$.

While the estimate of one minus true positives at a threshold t is similar in spirit to our number of unlabeled examples in the top bin and the estimate of one minus true negatives at a threshold t is similar in spirit to our number of positive examples in the unlabeled data, the functional form of these estimates are very different. Scott (2015) estimator is the ratio of quantities obtained by binomial tail inversion (i.e. upper bound in the numerator and lower bound in the denominator). By contrast, the final BBE estimate is simply the ratio of empirical CDFs at the optimal threshold. Mathematically, we have

$$\widehat{w}\alpha_{\text{Scott}} = \frac{\widehat{w}q_u(c_{\text{Scott}}) + \text{binv}(n_u, \widehat{w}q_u(c_{\text{Scott}}), \delta/n_u)}{\widehat{w}q_p(c_{\text{Scott}}) - \text{binv}(n_p, \widehat{w}q_p(c_{\text{Scott}}), \delta/n_p)} \quad \text{and} \tag{C.29}$$

$$\widehat{w}\alpha_{\text{BBE}} = \frac{\widehat{w}q_u(c_{\text{BBE}})}{\widehat{w}q_p(c_{\text{BBE}})}\,, \tag{C.30}$$

where $c_{\text{Scott}} = \arg\min_{c\in[0,1]} \frac{\widehat{w}q_u(c)+\text{binv}(n_u,\widehat{w}q_u(c),\delta/n_u)}{\widehat{w}q_p(c)-\text{binv}(n_p,\widehat{w}q_p(c),\delta/n_p)}$ and $\text{binv}(n_p, q_p(c), \delta/n_p)$ is the tightest possible deviation bound for a binomial random variable (Scott, 2015) and and c_{BBE} is given by Algorithm 17. Moreover, Scott (2015) provide no theoretical guarantees for their

Dataset	Model	$(\text{TED})^n$	BBE*	DEDPUL*	Scott*
Binarized CIFAR	ResNet	**0.018**	0.072	0.075	0.091
CIFAR Dog vs Cat	ResNet	**0.074**	0.120	0.113	0.158
Binarized MNIST	MLP	**0.021**	0.028	0.027	0.063
MNIST17	MLP	**0.003**	0.008	0.006	0.037

Table C.1: Absolute estimation error when α is 0.5. "*" denote oracle early stopping as defined in Sec. 2.6. As mentioned in Scott (2015) implementation in https://web.eecs.umich.edu/~cscott/code/mpe_v2.zip, we use the binomial inversion at δ instead of δ/n (rescaling using the union bound). Since we are using Binomial inversion at n discrete points simultaneously, we should use the union-bound penalty. However, using union bound penalty substantially increases the bias in their estimator.

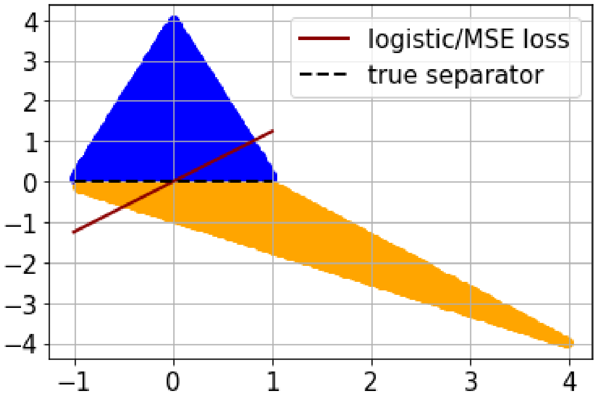
heuristic estimator $\widehat{w}\alpha_{\text{Scott}}$. On the hand, we provide guarantees that our estimator $\widehat{w}\alpha_{\text{BBE}}$ will converge to the best estimate achievable over all choices of the bin size and provide consistent estimates whenever a pure top bin exists. Supporting theoretical results of BBE, we observe that these choices in BBE create substantial differences in the empirical performance as observed in Table C.1. We repeat experiment for MPE from Sec. 2.6 where we compare other methods with the Scott (2015) estimator as defined in (C.29).

As a side note, a naive implementation of $\widehat{w}\alpha_{\text{Scott}}$ instead of (C.29) where we directly minimize the empirical ratio yields poor estimates due to noise introduced with finite samples. In our experiments, we observed that $\widehat{w}\alpha_{\text{Scott}}$ improves a lot over this naive estimator.

C.3 Toy setup

Jain et al. (2016) and Ivanov (2019) discuss Bayes optimality of the PvU classifier (or its one-to-one mapping) as a sufficient condition to preserve α in transformed space. However, in a simple toy setup (in App. C.3), we show that even when the hypothesis class is well specified for PvN learning, it will not in general contain the Bayes optimal scoring function for PvU data and thus PvU training will not recover the Bayes-optimal scoring function, even in population.

Consider a scenario with $\mathcal{X} = \mathbb{R}^2$. Assume points from the positive class are sampled uniformly from the interior of the triangle defined by coordinates $\{(-1, 0.1), (0, 4), (1, 0.1)\}$ and negative points are sampled uniformly from the interior of triangle defined by coordinates $\{(-1, -0.1), (4, -4), (1, -0.1)\}$. Ref. to Fig. C.1 for a pictorial representation. Let mixture proportion be 0.5 for the unlabeled data. Given access to distribution of positive data and unlabeled data, we seek to train a linear classifier to minimize logistic or Brier loss for PvU



(a)

Figure C.1: Blue points show samples from the positive distribution and orange points show samples from the negative distribution. Unalabeled data is obtained by mixing positive and negative distribution with equal proportion. BCE (or Brier) loss minimization on P vs U data leads to a classifiers that is not consistent with the ranking of the Bayes optimal score function.

training.

Since we need a monotonic transformation of the Bayes optimal scoring function, we want to recover a predictor parallel to x-axis, the Bayes optimal classifier for PvN training. However, minimizing the logistic loss (or Brier loss) using numerical methods, we obtain a predictor that is inclined at a non-zero acute angle to the x-axis. Thus, the PvU classifier obtained fails to satisfy the sufficient condition from Jain et al. (2016) and Ivanov (2019). On the other hand, note that the linear classifier obtained by PvU training satisfies the pure positive bin property.

Now we show that under the subdomain assumption (Ramaswamy et al., 2016; Scott, 2015), any monotonic transformation of Bayes optimal scoring function induces positive pure bin property. First, we define the subdomain assumption.

Assumption 5 (Subdomain assumption)**.** *A family of subsets $\mathcal{S} \subseteq 2^{\mathcal{X}}$, and distributions p_p, p_n are said to satisfy the anchor set condition with margin $\gamma > 0$, if there exists a compact set $A \in \mathcal{S}$ such that $A \subseteq \text{supp}(p_p)/\text{supp}(p_n)$ and $p_p(A) \geqslant \gamma$.*

Note that any monotonic mapping of the Bayes optimal scoring function can be represented by $\tau' = g \circ \tau$, where g is a monotonic function and

$$\tau(x) = \begin{cases} p_p(x)/p_u(x) & \text{if } p_p(x) > 0 \\ 0 & \text{o.w.} \end{cases} \tag{C.31}$$

For any point $x \in A$ and $x' \in \mathcal{X}/A$, we have $\tau(x) > \tau(x')$ which implies $\tau'(x) > \tau'(x')$. Thus, any monotonic mapping of Bayes optimal scoring function yields the positive pure bin property with $\epsilon_p \geqslant \gamma$.

C.4 Analysis of CVIR

First we analyse our loss function in the scenario when the support of positives and negatives is separable. We assume that the true alpha α is known and we have access to populations of positive and unlabeled data. We also assume that their exists a separator $f^* : \mathcal{X} \mapsto \{0, 1\}$ that can perfectly separate the positive and negative distribution, i.e., $\int dx p_p(x)\mathbb{I}\left[f^*(x) \neq 1\right] + \int dx p_n(x)\mathbb{I}\left[f^*(x) \neq 0\right] = 0$. Our learning objective can be written as jointly optimizing a classifier f and a weighting function w on the unlabeled distribution:

$$\begin{aligned} &\min_{f \in \mathcal{F}, w} \int dx p_p(x) l(f(x), 1) + \frac{1}{1-\alpha} \int dx p_u(x) w(x) l(f(x), 0)\,, \\ &\text{s.t. } w : \mathcal{X} \mapsto [0, 1]\,, \int dx p_u(x) w(x) = 1 - \alpha\,. \end{aligned} \tag{C.32}$$

The following proposition shows that minimizing the objective (C.32) on separable positive and negative distributions gives a perfect classifier.

Proposition C.4.1. *For $\alpha \in (0, 1)$, if there exists a classifier $f^* \in \mathcal{F}$ that can perfectly separate the positive and negative distributions, optimizing objective (C.32) with 0-1 loss leads to a classifier f that achieves 0 classification error on the unlabeled distribution.*

Proof. First we observe that having $w(x) = 1 - f^*(x)$ leads to the objective value being minimized to 0 as well as a perfect classifier f. This is because

$$\frac{1}{1-\alpha} \int dx p_u(x)(1 - f^*(x)) l(f(x), 0) = \int dx p_n(x) l(f(x), 0)$$

thus the objective becomes classifying positive v.s. negative, which leads to a perfect classifier if $\mathcal{F}$ contains one. Now we show that for any f such that the classification error is non-zero then the objective (C.32) must be greater than zero no matter what w is. Suppose f satisfies

$$\int dx p_p(x) l(f(x), 1) + \int dx p_n(x) l(f(x), 0) > 0\,.$$

We know that either $\int dx p_p(x) l(f(x), 1) > 0$ or $\int dx p_n(x) l(f(x), 0) > 0$ will hold. If $\int dx p_p(x) l(f(x), 1) > 0$ we know that (C.32) must be positive. If $\int dx p_p(x) l(f(x), 1) = 0$ and $\int dx p_n(x) l(f(x), 0) > 0$ we have $l(f(x), 0) = 1$ almost everywhere in $p_p(x)$ thus

$$\begin{aligned} &\frac{1}{1-\alpha} \int dx p_u(x) w(x) l(f(x), 0) \\ &= \frac{\alpha}{1-\alpha} \int dx p_p(x) w(x) l(f(x), 0) + \int dx p_n(x) w(x) l(f(x), 0) \\ &= \frac{\alpha}{1-\alpha} \int dx p_p(x) w(x) + \int dx p_n(x) w(x) l(f(x), 0)\,. \end{aligned}$$

If $\int dx p_p(x) w(x) > 0$ we know that (C.32) must be positive. If $\int dx p_p(x) w(x) = 0$, since we know that

$$\int dx p_u(x) w(x) = \alpha \int dx p_p(x) w(x) + (1-\alpha) \int dx p_n(x) w(x) = 1 - \alpha$$

we have $\int dx p_n(x) w(x) = 1$ which means $w(x) = 1$ almost everywhere in $p_n(x)$. This leads to the fact that $\int dx p_n(x) l(f(x), 0) > 0$ indicates $\int dx p_n(x) w(x) l(f(x), 0) > 0$, which concludes the proof.

□

The intuition is that, any classifier that discards an $\widetilde{\alpha} > 0$ proportion of negative distribution from unlabeled will have loss strictly greater than zero with our CVIR objective. Since only a perfect linear separator (with weights → ∞) can achieves loss → 0, CVIR objective will (correctly) discard the α proportion of positive from unlabeled data achieving a classifier that perfectly separates the data.

We leave theoretic investigation on non-separable distributions for future work. However, as an initial step towards a general theory, we show that in the population case one step of our alternating procedure cannot increase the loss.

Consider the following objective function

$$\begin{aligned} L(f_t, w_t) &= E_{x \sim P_p}[l(f_t(x), 0)] + E_{x \sim P_u}[w_t(x) l(f_t(x), 1)] \\ \text{such that} \qquad & E_{x \sim P_u}[w(x)] = 1 - \alpha \text{ and } w(x) \in \{0, 1\} \end{aligned} \tag{C.33}$$

Given f_t and w_t, CVIR can be summarized as the following two step iterative procedure: (i) Fix f_t, optimize the loss to obtain w_{t+1}; and (ii) Fix w_{t+1} and optimize the loss to obtain f_{t+1}. By construction of CVIR, we select w_{t+1} such that we discard points with highest loss, and hence $L(f_t, w_{t+1}) \leqslant L(f_t, w_t)$. Fixing w_{t+1}, we minimize the $L(f_t, w_{t+1})$ to obtain f_{t+1} and hence $L(f_{t+1}, w_{t+1}) \leqslant L(f_t, w_{t+1})$. Combining these two steps, we get $L(f_{t+1}, w_{t+1}) \leqslant L(f_t, w_t)$.

C.5 Prior Method Details

PU learning Next, we briefly discuss recent methods for MPE that operate in the classifier output space to avoid curse of dimensionality:

(i) **EN:** Given a domain discriminator classifier f_d trained to discriminate between positive and unlabeled, Elkan and Noto (2008) proposed the following estimator: $\sum_{x_i \in X_p} f_d(x_i) / \sum_{x_i \in X_u} f_d(x_i)$ where X_p is the set of positive examples and X_u is the set of unlabeled examples.

(ii) **DEDPUL:** Given a domain discriminator classifier f_d, Ivanov (2019) proposed an estimator that leverages density of the data in the output space of the classifier f_d to directly estimate $\min p_u(f(x))/p_p(f(x))$.

(iii) **BBE:** BBE (Garg et al., 2021b) identifies a threshold on probability scores assigned by the classifier f_d such that by estimating the ratio between the fractions of positive and unlabeled points receiving scores above the threshold, we obtain proportion of positives in unlabeled.

After obtaining an estimate for mixture proportion α, following methods can be employed for PU classification:

(i) **Domain Discriminator:** Given positive and unlabeled data, Elkan and Noto (2008) trained a classifier f_d to discriminator between them. To make a prediction on test point from unlabeled data, we can then use Bayes rule to obtain the following transformation on probabilistic output of the domain discriminator: $f = \alpha \left(\frac{m}{n}\right) \left(\frac{f_d(x)}{1-f_d(x)}\right)$, where n and m are the number of positives and unlabeled examples used to train f_d (Elkan and Noto, 2008).

(ii) **uPU:** Du Plessis et al. (2015) proposed an unbiased loss estimator for positive versus negative training. In particular, since $p_u = \alpha p_p + (1-\alpha) p_n$, the loss on negative examples $\mathbb{E}_{p_n}\left[\ell(f(x); -1)\right]$ can be estimated as:

$$\mathbb{E}_{p_n}\left[\ell(f(x); -1)\right] = \frac{1}{1-\alpha}\left[\mathbb{E}_{p_u}\left[\ell(f(x); -1)\right] - \alpha \mathbb{E}_{p_p}\left[\ell(f(x); -1)\right]\right] . \tag{C.34}$$

Thus, a classifier can be trained with the following uPU loss:

$$\mathcal{L}_{\text{uPU}}(f) = \alpha \mathbb{E}_{p_p}\left[\ell(f(x); +1)\right] + \mathbb{E}_{p_u}\left[\ell(f(x); -1)\right] - \alpha \mathbb{E}_{p_p}\left[\ell(f(x); -1)\right] . \tag{C.35}$$

(iii) **nnPU:** While unbiased losses exist that estimate the PvN loss given PU data and the mixture proportion α, this unbiasedness only holds before the loss is optimized, and becomes ineffective with powerful deep learning models capable of memorization. Kiryo et al. (2017) proposed the following non-negative regularization for unbiased PU learning:

$$\mathcal{L}_{\text{nnPU}}(f) = \alpha \mathbb{E}_{p_p}\left[\ell(f(x); +1)\right] + \max\left\{\mathbb{E}_{p_u}\left[\ell(f(x); -1)\right] - \alpha \mathbb{E}_{p_p}\left[\ell(f(x); -1)\right], 0\right\} . \tag{C.36}$$

(iv) **CVIR:** Garg et al. (2021b) proposed CVIR objective, which discards the highest loss α fraction of unlabeled examples on each training epoch, removing the incentive to overfit to the unlabeled positive examples. CVIR loss is defined as

$$\mathcal{L}_{\text{CVIR}}(f) = \alpha \mathbb{E}_{p_p}\left[\ell(x, 1; f)\right] + \mathbb{E}_{p_u}\left[w(x)\ell(x, -1; f)\right] , \tag{C.37}$$

where weights $w(x) = \mathbb{I}\left[\ell(x, -1; f) \leqslant \text{VIR}_\alpha(f)\right]$ for $\text{VIR}_\alpha(f)$ defined as $\text{VIR}_\alpha(f) = \inf\{\tau \in \mathbb{R} : \text{P}_{x \sim p_u}(\ell(x, -1; f) \leqslant \tau) \geqslant 1-\alpha\}$. Intuitively, $\text{VIR}_\alpha(f)$ identifies a threshold τ to capture bottom $1-\alpha$ fraction of the loss $\ell(x, -1)$ for points x sampled from p_u.

C.6 Experimental Details

Below we present dataset details. We present experiments with MNIST Overlap in App. C.7.7.

Dataset	Simula7ed PU Dataset	P vs N	#Positives		#Unlabeled	
			Train	Val	Train	Val
CIFAR10	Binarized CIFAR	[0-4] vs [5-9]	12500	12500	2500	2500
	CIFAR Dog vs Cat	3 vs 5	2500	2500	500	500
MNIST	Binarized MNIST	[0-4] vs [5-9]	15000	15000	2500	2500
	MNIST 17	1 vs 7	3000	3000	500	500
	MNIST Overlap	[0-7] vs [3-9]	150000	15000	2500	2500
IMDb	IMDb	pos vs neg	6250	6250	5000	5000

For CIFAR dataset, we also use the standard data augementation of random crop and horizontal flip. PyTorch code is as follows:

```
(transforms.RandomCrop(32, padding=4),
transforms.RandomHorizontalFlip())
```

C.6.1 Architecture and Implementation Details

All experiments were run on NVIDIA GeForce RTX 2080 Ti GPUs. We used PyTorch (Paszke et al., 2019) and Keras with Tensorflow (Abadi et al., 2016) backend for experiments.

For CIFAR10, we experiment with convolutional nets and MLP. For MNIST, we train MLP. In particular, we use ResNet18 (He et al., 2016) and all convolution net (Springenberg et al., 2014) . Implementation adapted from: https://github.com/kuangliu/pytorch-cifar.git. We consider a 4-layered MLP. The PyTorch code for 4-layer MLP is as follows:

```
 nn.Sequential(nn.Flatten(),
nn.Linear(input_dim, 5000, bias=True),
nn.ReLU(),
nn.Linear(5000, 5000, bias=True),
nn.ReLU(),
nn.Linear(5000, 50, bias=True),
nn.ReLU(),
nn.Linear(50, 2, bias=True)
)
```

For all architectures above, we use Xaviers initialization (Glorot and Bengio, 2010). For all methods except nnPU and uPU, we do cross entropy loss minimization with SGD optimizer with momentum 0.9. For convolution architectures we use a learning rate of 0.1 and MLP architectures we use a learning rate of 0.05. For nnPU and uPU, we minimize sigmoid loss with ADAM optimizer with learning rate 0.0001 as advised in its original paper. For all methods, we fix the weight decay param at 0.0005.

For IMDb dataset, we fine-tune an off-the-shelf uncased BERT model (Devlin et al., 2019). Code adapted from Hugging Face Transformers (Wolf et al., 2020): `https://huggingface.co/transformers/v3.1.0/custom_datasets.html`. For all methods except nnPU and uPU, we do cross entropy loss minimization with Adam optimizer with learning rate 0.00005 (default params). With the same hyperparameters and Sigmoid loss, we could not train BERT with nnPU and uPU due to vanishing gradients. Instead we use learning rate 0.00001.

C.6.2 Division between training set and hold-out set

Since the training set is used to learn the classifier (parameters of a deep neural network) and the hold-out set is just used to learn the mixture proportion estimate (scalar), we use a larger dataset for training. Throughout the experiments, we use an 80-20 split of the original set.

At a high level, we have an error bound on the mixture proportion estimate and we can use that to decide the split in general. As long as we use enough samples to make the $\mathcal{O}(1/\sqrt{n})$ small in our bound in Theorem 4.4.1, we can use the rest of the samples to learn the classifier.

C.7 Additional Experiments

C.7.1 nnPU vs PN classification

In this section, we compare the performance of nnPU and PvN training on the same positive and negative (from the unlabeled) data at $\alpha = 0.5$. We highlight the huge classification performance gap between nnPU and PvN training and show that training with CVuO objective partially recovers the performance gap. Note, to train PvN classifier, we use the same hyperparameters as that with PvU training.

C.7.2 Under-Fitting due to pessimistic early stopping

Ivanov (2019) explored the following heuristics for ad-hoc early stopping criteria: training proceeds until the loss on unseen PU data ceases to decrease. In particular, the authors suggested early stopping criterion based on the loss on unseen PU data doesn't decrease in epochs separated by a pre-defined window of length l. The early stopping is done when this happens consecutively for l epochs. However, this approach leads to severe under-fitting. When we fix $l = 5$, we observe a significant performance drop in CIFAR classification and MPE.

With PvU training, the performance of ResNet model on Binarized CIFAR (in Table 4.2) drops from 78.3 (orcale stopping) to 60.4 (with early stopping). Similar on CIFAR CAT

Dataset	Model	nnPU (known α)	PvN	CVuO (known α)	(TED)^n (unknown α)
Binarized CIFAR	ResNet	76.8	86.9	82.6	82.7
	All Conv	72.1	76.7	77.1	76.8
	MLP	63.9	65.1	65.9	63.2
CIFAR Dog vs Cat	ResNet	72.6	80.4	74.0	76.1
	All Conv	68.4	77.9	71.0	72.2
Binarized MNIST	MLP	95.9	96.7	96.4	95.9
MNIST17	MLP	98.2	99.0	98.6	98.6
IMDb	BERT	86.2	89.1	87.4	88.1

Table C.2: Accuracy for PvN classification with nnPU, PvN, CVuO objective and (TED)^n training. Results reported by aggregating aggregating over 10 epochs.

vs Dog, the performance of the same architecture drops from 71.6 (orcale stopping) to 58.4 (with early stopping). Note that the decrease in accuracy is less or not significant for MNIST. With PvU training, the performance of MLP model on Binarized MNIST (in Table 4.2) drops from 94.5 (orcale stopping) to 94.1 (with early stopping). This is because we obtain good performance on MNIST early in training.

C.7.3 Overfitting on unlabeled data as PvU training proceeds

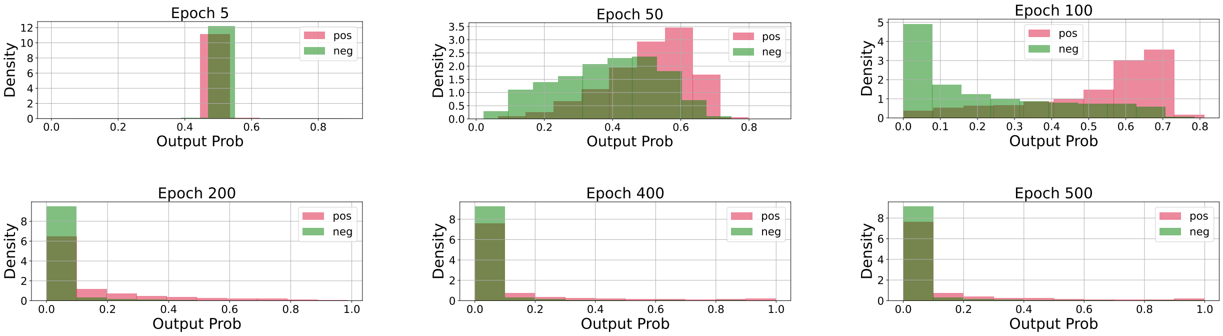


Figure C.2: Score assigned by the classifier to positive and negative points in the unlabeled training set as PvU training proceeds. As training proceeds, classifier memorizes both positive and negative in unlabeled as negatives.

In Fig. C.2, we show the distribution of unlabeled training points. We show that as positive versus unlabeled training proceeds with a ResNet-18 model on binarized CIFAR dataset, classifier memorizes all the unlabeled data as negative assigning them very small scores (i.e., the probability of them being negative).

C.7.4 Ablations to (TED)^n

Varying the number of warm start epochs We now vary the number of warm start epochs with (TED)^n. We observe that increasing the number of warm start epochs doesn't hurt (TED)^n even when the classifier at the end of the warm start training memorized PU training data due PvU training. While in many cases (TED)^n training without warm start is able to recover the same performance, it fails to learn anything for CIFAR Dog vs Cat with all convolutional neural network. This highlights the need for warm start training with (TED)^n.

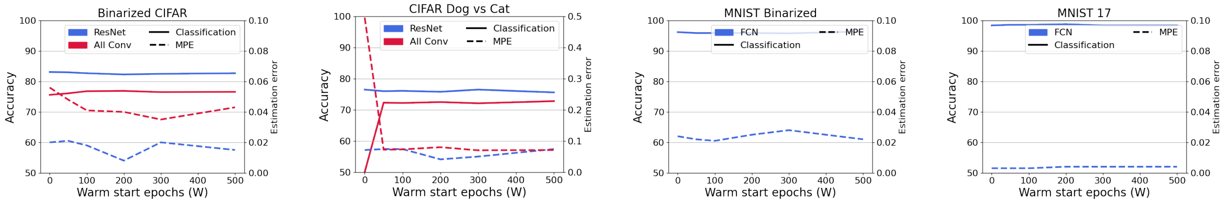


Figure C.3: Classification and MPE results with varying warm start epochs W with (TED)^n

Varying the true mixture proportion α Next, we vary α, the true mixture proportion and present results for MPE and classification in Fig. C.4. Overall, across all α, our method (TED)^n is able to achieve superior performance as compared to alternate algorithms. We omit high α for CIFAR and IMDb datasets as all the methods result in trivial accuracy and mixture proportion estimate.

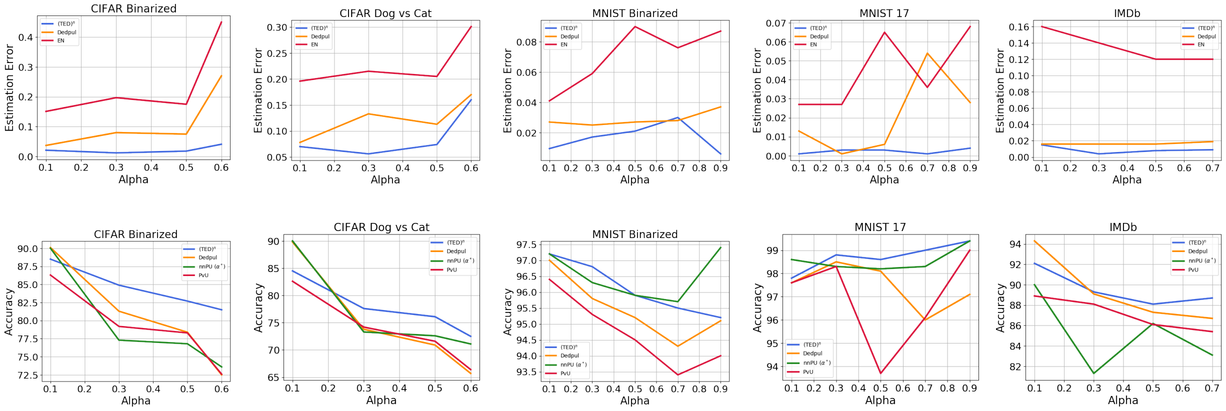


Figure C.4: MPE and Classification results with varying mixture proportion. For each method we show results with the best performing architecture.

C.7.5 Classification and MPE results with error bars

Dataset	Model	$(\text{TED})^n$	BBE*	DEDPUL*	EN	KM2	TiCE
Binarized CIFAR	ResNet	**0.026 ± 0.005**	0.091 ± 0.027	0.091 ± 0.023	0.192 ± 0.007		
	All Conv	0.042 ± 0.003	**0.037 ± 0.018**	0.052 ± 0.017	0.221 ± 0.017	0.168 ± 0.207	0.194 ± 0.039
	MLP	0.225 ± 0.013	0.177 ± 0.011	**0.138 ± 0.009**	0.372 ± 0.002		
CIFAR Dog vs Cat	ResNet	**0.078 ± 0.010**	0.176 ± 0.015	0.170 ± 0.010	0.226 ± 0.003	0.331 ± 0.238	0.286 ± 0.013
	All Conv	**0.066 ± 0.015**	0.128 ± 0.020	0.115 ± 0.014	0.250 ± 0.019		
Binarized MNIST	MLP	**0.024 ± 0.001**	0.032 ± 0.001	0.031 ± 0.003	0.080 ± 0.009	0.029 ± 0.008	0.056 ± 0.05
MNIST17	MLP	**0.003 ± 0.000**	0.023 ± 0.017	0.021 ± 0.011	0.028 ± 0.017	0.022 ± 0.003	0.043 ± 0.023
IMDb	BERT	**0.008 ± 0.001**	0.011 ± 0.002	0.016 ± 0.005	0.07 ± 0.01	-	-

Table C.3: Absolute estimation error when α is 0.5. "*" denote oracle early stopping as defined in Sec. 2.6. Results reported by aggregating absolute error over 10 epochs and 3 seeds.

C.7.6 Experiments on UCI dataset

In this section, we will present results on 5 UCI datasets.

Dataset	#Positives		#Unlabeled	
	Train	Val	Train	Val
concrete	162	162	81	81
mushroom	1304	1304	652	652
landsat	946	946	472	472
pageblock	185	185	92	92
spambase	604	604	302	302

We train a MLP with 2 hidden layers each with 512 units. The PyTorch code for 4-layer MLP is as follows:

```
 nn.Sequential(nn.Flatten(),
nn.Linear(input_dim, 512, bias=True),
nn.ReLU(),
nn.Linear(512, 512, bias=True),
nn.ReLU(),
nn.Linear(512, 2, bias=True),
)
```

Similar to vision datasets and architectures, we do cross entropy loss minimization with SGD optimizer with momentum 0.9 and learning rate 0.1. For nnPU and uPU, we minimize

Dataset	Model	$(\text{TED})^n$ (unknown α)	CVIR (known α)	PvU* (known α)	DEDPUL* (unknown α)	nnPU (known α)	uPU* (known α)
Binarized CIFAR	ResNet	$\mathbf{82.7 \pm 0.13}$	82.3 ± 0.18	76.9 ± 1.12	77.1 ± 1.52	77.2 ± 1.03	76.7 ± 0.74
	All Conv	77.9 ± 0.29	$\mathbf{78.1 \pm 0.47}$	75.8 ± 0.75	77.1 ± 0.64	73.4 ± 1.31	72.5 ± 0.21
	MLP	64.2 ± 0.37	$\mathbf{66.9 \pm 0.28}$	61.6 ± 0.38	62.6 ± 0.30	63.1 ± 0.79	64.0 ± 0.24
CIFAR Dog vs Cat	ResNet	$\mathbf{75.2 \pm 1.74}$	73.3 ± 0.94	67.3 ± 1.52	67.0 ± 1.46	71.8 ± 0.33	68.8 ± 0.53
	All Conv	$\mathbf{73.0 \pm 0.81}$	71.7 ± 0.47	70.5 ± 0.60	69.2 ± 0.86	67.9 ± 0.52	67.5 ± 2.28
Binarized MNIST	MLP	95.6 ± 0.42	$\mathbf{96.3 \pm 0.07}$	94.2 ± 0.58	94.8 ± 0.10	96.1 ± 0.14	95.2 ± 0.19
MNIST17	MLP	$\mathbf{98.7 \pm 0.25}$	$\mathbf{98.7 \pm 0.09}$	96.9 ± 1.51	97.7 ± 0.62	98.4 ± 0.20	98.4 ± 0.09
IMDb	BERT	$\mathbf{87.6 \pm 0.20}$	87.4 ± 0.25	86.1 ± 0.53	87.3 ± 0.18	86.2 ± 0.25	85.9 ± 0.12

Table C.4: Accuracy for PvN classification with PU learning. "*" denote oracle early stopping as defined in Sec. 2.6. Results reported by aggregating over 10 epochs and 3 seeds.

sigmoid loss with ADAM optimizer with learning rate 0.0001 as advised in its original paper. For all methods, we fix the weight decay param at 0.0005.

Dataset	$(\text{TED})^n$	BBE*	DEDPUL*	EN*	KM2	TiCE
concrete	**0.071**	0.152	0.176	0.239	0.099	0.268
mushroom	**0.001**	0.015	0.014	0.013	0.038	0.069
landsat	0.022	0.021	**0.012**	0.080	0.037	0.027
pageblock	**0.007**	0.066	0.041	0.135	0.008	0.298
spambase	**0.006**	0.047	0.077	0.127	0.062	0.276

Table C.5: Absolute estimation error when α is 0.5. "*" denote oracle early stopping as defined in Sec. 2.6. Results reported by aggregating absolute error over 10 epochs.

On 4 out of 5 UCI datasets, our proposed methods are better than the best performing alternatives (Table C.5 and Table C.6).

C.7.7 Experiments on MNIST Overlap

Similar to binarized MNIST, we create a new dataset called MNIST Overlap, where the positive class contains digits from 0 to 7 and the negative class contains digits from 3 to 9. This creates a dataset with overlap between positive and negative support. Note that while the supports overlap, we sample images from the overlap classes with replacement, and hence, in absence of duplicates in the dataset, exact same images don't appear both in positive and negative subsets.

Dataset	$(\text{TED})^n$ (unknown α)	CVuO (known α)	PvU* (known α)	DEDPUL* (unknown α)	nnPU (known α)	uPU* (known α)
concrete	**86.3**	80.1	83.1	83.7	83.2	84.4
mushroom	96.4	96.3	**98.7**	**98.7**	97.5	93.9
landsat	**93.8**	93.1	93.4	92.4	92.9	92.3
pageblock	**95.7**	**95.7**	95.1	94.5	93.9	93.9
spambase	**89.4**	88.1	89.2	86.8	88.5	87.7

Table C.6: Accuracy for PvN classification with PU learning. "*" denote oracle early stopping as defined in Sec. 2.6. Results reported by aggregating aggregating over 10 epochs.

We train MLP with the same hyperparameters as before. Our findings in Table C.7 and Table C.8 highlight superior performance of the proposed approaches in the cases of support overlap.

Dataset	$(\text{TED})^n$	BBE*	DEDPUL*	EN*	KM2	TiCE
MNIST Overlap	**0.035**	0.100	0.104	0.196	0.099	0.074

Table C.7: Absolute estimation error when α is 0.5. "*" denote oracle early stopping as defined in Sec. 2.6. Results reported by aggregating absolute error over 10 epochs.

Dataset	$(\text{TED})^n$ (unknown α)	CVuO (known α)	PvU* (known α)	DEDPUL* (unknown α)	nnPU (known α)	uPU* (known α)
MNIST Overlap	**79.0**	78.4	77.4	77.5	78.6	78.8

Table C.8: Accuracy for PvN classification with PU learning. "*" denote oracle early stopping as defined in Sec. 2.6. Results reported by aggregating aggregating over 10 epochs.

Appendix D

Appendix: Domain Adaptation Under Open Set Label Shift

D.1 Reduction of OSLS into k PU problems

Under the strong positivity condition, the OSLS problem can be broken down into k PU problems as follows: By treating a given source class $y_j \in \mathcal{Y}_s$ as *positive* and grouping all other classes together as *negative* we observe that the unlabeled target data is then a mixture of data from the positive and negative classes. This yields a PU learning problem and the corresponding mixture proportion gives the fraction α_j of class y_j among the target data. By iterating this process for all source classes, we can solve for the entire target label marginal $p_t(y)$. Thus, OSLS reduces to k instances of PU learning problem. Formally, note that $p_t(x)$ can be written as:

$$p_t(x) = \underbrace{p_t(y = j)}_{\alpha_j} \underbrace{p_s(x|y = j)}_{p_p} + (1 - p_t(y = j)) \underbrace{\left(\sum\nolimits_{i \in \mathcal{Y} \setminus \{j\}} \frac{p_t(y = i)}{1 - p_t(y = j)} p_s(x|y = i) \right)}_{p_n}, \quad \text{(D.1)}$$

individually for all $j \in \mathcal{Y}_s$. By repeating this reduction for all classes, we obtain k separate PU learning problems. Hence, a natural choice is to leverage this structure and solve k PU problems to solve the original OSLS problem.

In particular, for each class $j \in \mathcal{Y}_s$, we can first estimate its prevalence $\widehat{\alpha}_j$ in the unlabeled target. Then the target marginal for the novel class is given by $\widehat{\alpha}_{k+1} = 1 - \sum_{i=1}^{k} \widehat{\alpha}_i$. For classification, we can train k PU learning classifiers f_i, where f_i is trained to classify a source class i versus others in target. Assuming that each f_j returns a score between $[0, 1]$, during test time, an example x is classified as $f(x)$ given by

$$f(x) = \begin{cases} \arg\max_{j \in \mathcal{Y}_s} f_j(x) & \text{if } \max_{j \in \mathcal{Y}_s} f_j(x) \geqslant 0.5 \\ k+1 & \text{o.w.} \end{cases} \quad \text{(D.2)}$$

That is, if each classifier classifies the example as belonging to other in unlabeled, then we classify the example as belonging to the class $k+1$. In our main experiments, to estimate α_j and to train f_j classifiers for all $j \in \mathcal{Y}_s$, we use BBE and CVIR as described before which was shown to outperform alternative approaches in Garg et al. (2021b). We ablate with other methods in App. D.6.8.

Note that mathematically any OSLS problems can be thought of as k-PU problems as per (D.1). However, for identifiablity of each of these PU problems, we need the irreduciblity assumption (Bekker and Davis, 2020). Put simply, for individual PU problems defined for source classes $j \in \mathcal{Y}_s$, we need existence of a sub-domain X_j such that we only observe example for that class j in X_j. Collectively X_j gives us the X_{sp} defined in the strong positivity condition.

Failure due to error-accumulation While trading off bias with variance, PU learning algorithms tend to over-estimate the mixture proportion (Bekker and Davis, 2020; Garg et al., 2021b). This error incurred due to bias can be mild for a single mixture proportion estimation task but accumulates with increasing number of classes (i.e., k). This error accumulation can significantly under-estimate the proportion of novel class when estimated by subtracting the sum of prevalence of source classes in target from 1.

D.2 Proofs for identifiability of OSLS

For ease, we re-state Proposition 5.4 and Proposition 5.4.

[Necessary conditions] Assume $p_t(y) > 0$ for all $y \in \mathcal{Y}_t$. Then $p_t(y)$ is identified only if $p_t(x|y = k+1)$ and $p_s(x|y)$ for all $y \in \mathcal{Y}_s$ satisfy weak positivity, i.e., there must exists a subdomain $X_{\text{wp}} \subset X$ such that:

(i) $p_t(X_{\text{wp}}|y = k+1) = 0$; and

(ii) the matrix $[p_s(x|y)]_{x \in X_{\text{wp}}, y \in \mathcal{Y}_s}$ is full column-rank.

Proof. We prove this by contradiction. Assume that there exists a unique solution $p_t(y)$. We will obtain contradiction when both (i) and (ii) don't hold.

First, assume for no subset $X_{\text{wp}} \subseteq \mathcal{X}$, we have $[p_s(x|y)]_{x \in X_{\text{wp}}, y \in \mathcal{Y}_s}$ as full-rank. Then in that case, we have vectors $[p_s(x|y = j)]_{x \in \mathcal{X}}$ as linearly dependent for $j \in \mathcal{Y}_s$, i.e., there exists $[\alpha_j]_{j \in \mathcal{Y}_s} \in \mathbb{R}^k$ such that $\sum_j \alpha_j p_s(x|y = j) = 0$ for all $x \in \mathcal{X}$. Thus for small enough $\epsilon > 0$, we have infinite solutions of the form $[p_t(y = j) - \epsilon \cdot a_j]_{j \in \mathcal{Y}_s}$.

Hence, there exists $X_{\text{wp}} \subseteq \mathcal{X}$ for which we have $[p_s(x|y)]_{x \in X_{\text{wp}}, y \in \mathcal{Y}_s}$ as full-rank. Without loss of generality, we assume that $|X_{\text{wp}}| = k$. Assume that $p_t(X_{\text{wp}}|y = k+1) > 0$, i.e., $[p_t(x|y = k+1)]_{x \in X_{\text{wp}}}$ has $l < k$ zero entries. We will now construct another solution for the label marginal p_t. For simplicity we denote $A = [p_s(x|y)]_{x \in X_{\text{wp}}, y \in \mathcal{Y}_s}$. Consider the vector $v(\gamma) = [p_t(x) - (p_t(y = k+1) - \gamma) p_t(x|y = k+1)]_{x \in X_{\text{wp}}}$ for some $\gamma > 0$. Intuitively, when

$\gamma = 0$, we have $u = A^{-1}v(0)$ where $u = [p_t(y)]_{y \in \mathcal{Y}_s}$, i.e., we recover the true label marginal corresponding to source classes.

However, since the solution is not at vertex, there exists a small enough $\gamma > 0$ such that $u' = A^{-1}v(\gamma)$ with $\sum_j u'_j \leqslant 1$ and $u'_j \geqslant 0$. Since A is full-rank and $v(\gamma) \neq v(0)$, we have $u' \neq u$. Thus we construct a separate solution with u' as $[p_t(y)]_{y \in \mathcal{Y}_s}$ and $p_t(x) - \sum_{j \in \mathcal{Y}_s} u'_j p_s(x|y = j)$ as $p_t(x|y = k + 1)$. Hence, when there exists $X_{\text{wp}} \subseteq \mathcal{X}$ for which we have $[p_s(x|y)]_{x \in X_{\text{wp}}, y \in \mathcal{Y}_s}$ as full-rank, for uniqueness we obtain a contradiction on the assumption $p_t(X_{\text{wp}}|y = k + 1) > 0$. □

We now make some comments on the assumption $p_t(y) > 0$ for all $y \in \mathcal{Y}_t$ in Proposition 5.4. Since, $p_t(y)$ needs to satisfy simplex constraints, if the solution is at a vertex of simplex, then OSLS problem may not require weak positivity. For example, there exists contrived scenarios where $p_s(x|y = j) = p_s(x|y = k)$ for all $j, k \in \mathcal{Y}_s$ and $p_t(x|y = k+1) \neq p_s(x|y = j)$ for all $j \in \mathcal{Y}_s$. Then when $p_t(x) = p_t(x|y = k + 1)$, we can uniquely identify the OSLS solution even when weak positivity assumption is not satisfied.

[Sufficient conditions] The target marginal $p_t(y)$ is identified if for all $y \in \mathcal{Y} \backslash \{k + 1\}$, $p_t(x|y = k + 1)$ and $p_s(x|y)$ satisfy either:

(i) Strong positivity, i.e., there exists $X_{\text{sp}} \subset \mathcal{X}$ such that $p_t(X_{\text{sp}}|y = k + 1) = 0$ and the matrix $[p_s(x|y)]_{x \in X_{\text{sp}}, y \in \mathcal{Y}_s}$ is full-rank and diagonal; or

(ii) Separability, i.e., there exists $X_{\text{sep}} \subset \mathcal{X}$, such that $p_t(X_{\text{sep}}|y = k + 1) = 0$, $p_s(X_{\text{sep}}) = 1$, and the matrix $[p_s(x|y)]_{x \in X_{\text{sep}}, y \in \mathcal{Y}_s}$ is full column-rank.

Proof. For each condition, we will prove identifiability by constructing the unique solution.

Under strong positivity, for all $j \in \mathcal{Y}_s$ there exists $x \in X_{\text{sp}}$ such that $p_t(x|y = k) = 0$ for all $k \in \mathcal{Y}_t \backslash \{j\}$. Set $\alpha_j = \min_{x \in \mathcal{X}, p_s(x|y=j) > 0} \frac{p_t(x)}{p_s(x|y=j)}$, for all $j \in \mathcal{Y}_s$. For $x \in X_{\text{sp}}$ such that $p_t(x|y = k) = 0$ for all $k \in \mathcal{Y}_t \backslash \{j\}$, we get $\frac{p_t(x)}{p_s(x|y=j)} = p_t(y = j)$ and for all $x' \neq x$, we have $\frac{p_t(x)}{p_s(x|y=j)} \geqslant p_t(y = j)$. Thus, we get $\alpha_j = p_t(y = j)$. Finally, we get $\alpha_{k+1} = 1 - \sum_{j \in \mathcal{Y}_s} \alpha_j$. Plugging in values of the label marginal, we can obtain $p_t(x|y = k+1)$ as $p_t(x) - \sum_{y \in \mathcal{Y}_s} p_t(y = j) p_s(x|y = j)$.

Under separability, we can obtain the label marginal p_t for source classes by simply considering the set X_{sep}. Denote $A = [p(x|y)]_{x \in X_{\text{sep}}, y \in \mathcal{Y}_s}$ and $v = [p_t(x)]_{x \in X_{\text{sep}}}$. Then, since A is full column-rank by assumption, we can define $u = (A^T A)^{-1} A^T v$. For all $x \in X_{\text{sep}}$, we have $p_t(x) = \sum_{y \in \mathcal{Y}_s} p_t(y) p_s(x|y)$ and hence, $u = [p_t(y)]_{y \in \mathcal{Y}_s}$. Having obtained $[p_t(y)]_{y \in \mathcal{Y}_s}$, we recover $p_t(y = k + 1) = 1 - \sum_{j \in \mathcal{Y}_s} p_t(y = j)$ and $p_t(x|y = k + 1) = p_t(x) - \sum_{j \in \mathcal{Y}_s} p_t(y = j) p_s(x|y = j)$. □

D.2.1 Examples illustrating importance of weak positivity condition

In this section, we present two examples, one, to show that weak positivity isn't sufficient for identifiability. Second, we present another example where we show that conditions in Proposition 5.4 are not necessary for identifiability.

Example 1 Assume $\mathcal{X} = \{x_1, x_2, x_3, x_4, x_5\}$ and $\mathcal{Y}_t = \{1, 2, 3\}$. Suppose the $p_t(x|y = 1)$, $p_t(x|y = 2)$, and $p_t(x)$ are given as:

	$p_t(x\|y = 1)$	$p_t(x\|y = 2)$	$p_t(x)$
x_1	0.4	0.56	0.356
x_2	0.3	0.3	0.207
x_3	0.2	0.1	0.09
x_4	0.1	0.04	0.042
x_5	0.0	0.0	0.305

Here, there exists two separate $p_t(x|y = 3)$ and $p_t(y)$ that are consistent with the given $p_t(x|y = 1)$, $p_t(x|y = 2)$, and $p_t(x)$ and both the solutions satisfy weak positivity for two different X_{wp} and X'_{wp}.

In particular, notice that $p_t(x|y = 3) = [0.17, 0.0675, 0.0, 0.0, 0.7625]^T$ and $p_t(y) = [0.3, 0.3, 0.4]$ gives us the first solution. $p_t(x|y = 3) = [0.0, 0.0, 0.0645, 0.0096, 0.9839]^T$ and $p_t(y) = [0.19, 0.5, 0.31]$ gives us another solution. For solution 1, $X_{\text{wp}} = \{x_3, x_4\}$ and for solution 2, $X'_{\text{wp}} = \{x_1, x_2\}$. To check consistency of each solution notice that $\sum_{i\in\mathcal{Y}} p_t(y = i)p_t(x|y = i) = p_t(x)$ for each $x \in \mathcal{X}$. □

In the above example, the key is to show that absent knowledge of which x's constitute the set X_{wp}, we might be able to obtain multiple different solutions, each with different X_{wp} and both $p_t(y)$, $p_t(x|y = k + 1)$ satisfying the given information and simplex constraints.

Next, we will show that in certain scenarios weak positivity is enough for identifiability.

Example 2 Assume $\mathcal{X} = \{x_1, x_2, x_3, x_4\}$ and $\mathcal{Y}_t = \{1, 2, 3\}$. Suppose the $p_t(x|y = 1)$, $p_t(x|y = 2)$, and $p_t(x)$ are given as,

	$p_t(x\|y = 1)$	$p_t(x\|y = 2)$	$p_t(x)$
x_1	0.5	0.2	0.24
x_2	0.3	0.4	0.2
x_3	0.1	0.35	0.35
x_4	0.1	0.05	0.21

Here, out of all ${}^4C_2$ possibilities for X_{wp}, only one possibility yields a solution that satisfies weak positivity and simplex constraints. In particular, the solution is given by $p_t(x|y = 3) = [0.0, 0.0, 0.6, 0.4]^T$ and $p_t(y) = [0.4, 0.2, 0.4]$ with $X_{\text{wp}} = \{x_1, x_2\}$. □

In this example, we show that conditions in Proposition 5.4 are not necessary to ensure identifiability. For discrete domains, this example also highlights that we can check identifiability in exponential time for any OSLS problem given $p_t(x)$ and $p_s(x|y)$ for all $y \in \mathcal{Y}_s$.

D.2.2 Extending identifiability conditions to continuous distributions

To extend our identifiability conditions for continuous distributions, the linear independence conditions on the matrix $[p_s(x|y)]_{x \in X_{\text{sep}}, y \in \mathcal{Y}_s}$ has the undesirable property of being sensitive to changes on sets of measure zero. In particular, by changing a collection of linearly dependent distributions on a set of measure zero, we can make them linearly independent. As a consequence, we may impose a *stronger* notion of independence, i.e., the set of distributions $\{p(x|y) : y = 1, ..., k\}$ are such that there does not exist $v \neq 0$ for which $\int_X |\sum_y p(x|y) v_y| dx = 0$, where $X = X_{\text{wp}}$ for necessary condition and $X = X_{\text{sp}}$ for sufficiency. We refer this condition as *strict linear independence*.

D.3 PULSE Framework

In our PULSE framework, we build on top of BBE and CVIR from Garg et al. (2021b). Here, we elaborate on Step 3 and 5 in Algorithm 8.

Extending BBE algorithm to estimate target marginal among previously seen classes We first explain the intuition behind BBE approach. In a PU learning problem, given positive and unlabeled data, BBE estimates the fraction of positives in unlabeled in the push-forward space of the classifier. In particular, instead of operating in the original input space, BBE maps the inputs to one-dimensional outputs (i.e., a score between zero and one) which is the predicted probability of an example being from the positive class. BBE identifies a threshold on probability scores assigned by a domain discriminator classifier such that the ratio between the fractions of positive and unlabeled points receiving scores above the threshold is minimized. Intuitively, if their exists a threshold on probability scores assigned by the classifier such that the examples mapped to a score greater than the threshold are *mostly* positive, BBE aims to identify this threshold. Efficacy of BBE procedure relies on existence of such a threshold. This is referred to as the *top bin property*. We provide empirical evidence to the property in Fig. D.1 in App. D.4.1. We tailor BBE to estimate the relative fraction of previously seen classes in the target distribution by exploiting a k-way source classifier f_s trained on labeled source data. We describe the procedure in Algorithm 15.

We now introduce some notation needed to introduce the tailored BBE proceudre formally. For given probability density function p and a scalar output function f, define a function $q(z) = \int_{A_z} p(x) dx$, where $A_z = \{x \in \mathcal{X} : f(x) \geqslant z\}$ for all $z \in [0, 1]$. Intuitively, $q(z)$ captures the cumulative density of points in a top bin, the proportion of input domain that is assigned a value larger than z by the function f in the transformed space. We define an empirical estimator $\widehat{w}q(z)$ given a set $X = \{x_1, x_2, \ldots, x_n\}$ sampled iid from $p(x)$. Let $Z = f(X)$. Define $\widehat{w}q(z) = \sum_{i=1}^{n} \mathbb{I}\left[z_i \geqslant z\right]/n$.

Our modified BBE procedure proceeds as follows. Given a held-out dataset of source $\{\mathbf{X}_2^S, \mathbf{y}_2^S\}$ and unlabeled target samples $\mathbf{X}_2^T$, we push all examples through the source classifier f to obtain k dimensional outputs. For all $j \in \mathcal{Y}_s$, we repeat the following: Obtain $Z_s = f_j(\mathbf{X}_2^S[\mathrm{id}_j])$ and $Z_t = f_j(\mathbf{X}_2^T)$. Intuitively, Z_s and Z_t are the push forward mapping of the source classifier. Next, with Z_p and Z_u, we estimate $\widehat{w}q_s$ and $\widehat{w}q_t$. Finally, we estimate $[\widehat{w}p_t]_j$ as the ratio $\widehat{w}q_t(\widehat{w}c)/\widehat{w}q_s(\widehat{w}c)$ at $\widehat{w}c$ that minimizes the upper confidence bound at a pre-specified level δ and a fixed parameter $\gamma \in (0, 1)$. Our method is summarized in Algorithm 15. Throughout all the experiments, we fix δ at 0.1 and γ at 0.01.

Algorithm 15 Extending Best Bin Estimation (BBE) for Step 3 in Algorithm 8

```
input : Validation source $\{\mathbf{X}_2^S, \mathbf{y}_2^S\}$ and unlabeled target samples $\mathbf{X}_2^T$. Source classifier
    $f : \mathcal{X} \to \Delta^{k-1}$. Hyperparameter $0 < \delta, \gamma < 1$.
$\widehat{w}p_t \leftarrow$ zeros($size = |\mathcal{Y}_s|$)
for $j \in \mathcal{Y}_s$ do
    $\mathrm{id}_j \leftarrow$ where($\mathbf{y}_2^S = j$).
    $Z_s, Z_t \leftarrow \left[f(\mathbf{X}_2^S[\mathrm{id}_j])\right]_j, \left[f(\mathbf{X}_2^T)\right]_j$.
    $\widehat{w}q_s(z), \widehat{w}q_t(z) \leftarrow \frac{\sum_{z_i \in Z_s} \mathbb{I}[z_i \geqslant z]}{|\mathrm{id}_j|}, \frac{\sum_{z_i \in Z_t} \mathbb{I}[z_i \geqslant z]}{\left|\mathbf{X}_2^T\right|}$ for all $z \in [0, 1]$.
    $\widehat{w}c_j \leftarrow \arg\min_{c \in [0,1]} \left( \frac{\widehat{w}q_t(c)}{\widehat{w}q_s(c)} + \frac{1+\gamma}{\widehat{w}q_s(c)} \left( \sqrt{\frac{\log(4/\delta)}{2\left|\mathbf{X}_2^T\right|}} + \sqrt{\frac{\log(4/\delta)}{2|\mathrm{id}_j|}} \right) \right)$.
    $[\widehat{w}p_t]_j \leftarrow \frac{\widehat{w}q_t(\widehat{w}c_j)}{\widehat{w}q_s(\widehat{w}c_j)}$.
end for
output : Normalized target marginal among source classes $\widehat{w}p_t' \leftarrow \frac{\widehat{w}p_t}{\|\widehat{w}p_t\|1}$
```

Extending CVIR to train discriminator f_d and estimate novel class prevalence After estimating the fraction of source classes in target (i.e., $p_t'(j) = {p_t(y=j)}/{\sum_{k \in \mathcal{Y}_s} p_t(y=k)}$ for all $j \in \mathcal{Y}_s$), we re-sample the source data according to $p_t'(y)$ to mimic samples from distribution $p_s'(x)$. Thus, obtaining a PU learning problem instance, we resort to PU learning techniques to (i) estimate the fraction of novel class $p_t(y = k+1)$; and (ii) learn a binary classifier $f_d(x)$ to discriminate between label shift corrected source $p_s'(x)$ and novel class $p_t(x|y = k+1)$. Assume that sigmoid output $f_d(x)$ indicates predicted probability of an example x belonging to label shift corrected source $p_s'(x)$. With $\widehat{w}\mathcal{L}^+(f_\theta; X)$, we denote the loss incurred by f_θ when classifying examples from X as positive, i.e., $\widehat{w}\mathcal{L}^+(f_\theta; X) = \sum_{i=1}^{|X|} \frac{\ell(f_\theta(x_i), +1)}{|X|}$. Similarly, $\widehat{w}\mathcal{L}^-(f_\theta; X) = \sum_{i=1}^{|X|} \frac{\ell(f_\theta(x_i), -1)}{|X|}$

Given an estimate of the fraction of novel class $\widehat{w}p_t(y = k+1)$, CVIR objective creates a provisional set of novel examples $\mathbf{X}_1^N$ by removing $(1 - \widehat{w}p_t(y = k+1))$ fraction of examples from $\mathbf{X}_1^T$ that incur highest loss when predicted as novel class on each training epoch. Next, we update our discriminator f_d by minimizing loss on label shift corrected source

$\widetilde{\mathbf{X}}_1^S$ and provisional novel examples $\mathbf{X}_1^N$. This step is aimed to remove any incentive to overfit to the examples from $p'_s(x)$. Consequently, we employ the iterative procedure that alternates between estimating the prevalence of novel class $\widehat{w}p_t(y = k + 1)$ (with BBE) and minimizing the CVIR loss with estimated fraction of novel class. Algorithm 16 summarizes our approach which is used in Step 3 of Algorithm 8.

Note that we need to warm start with simple domain discrimination training, since in the initial stages mixture proportion estimate is often close to 1 rejecting all the unlabeled examples. In Garg et al. (2021b), it was shown that the procedure is not sensitive to the choice of number of warm start epochs and in a few cases with large datasets, we can even get away without warm start (i.e., $W = 0$) without hurting the performance. In our work, we notice that given an estimate $\widehat{\alpha}$ of prevalence of novel class, we can use unbiased PU error (C.35) on validation data as a surrogate to identify warm start epochs for domain discriminator training. In particular, we train the domain discriminator classifier for a large number of epochs, say $E(>> W)$, and then choose the discriminator, i.e., warm start epoch W at which f_d achieves minimum unbiased validation loss.

Finally, to obtain a $(k + 1)$-way classifier $f_t(x)$ on target we combine discriminator f_d and source classifier f_s with importance-reweighted label shift correction. In particular, for all $j \in \mathcal{Y}_s$, $[f_t(x)]_j = (f_d(x))\frac{w(j)\cdot[f_s(x)]_j}{\sum_{k\in\mathcal{Y}_s} w(k)\cdot[f_s(x)]_k}$ and $[f_t(x)]_{k+1} = 1 - f_d(x)$. Similarly, to obtain target marginal p_t, we re-scale the label shift estimate among previously seen classes with estimate of prevalence of novel examples, i.e., for all $j \in \mathcal{Y}_s$, assign $\widehat{w}p_t(y = j) = (1 - \widehat{w}p_t(y = k + 1)) \cdot \widehat{w}p'_t(y = j)$.

Overall, our approach proceeds as follows (Algorithm 8): First, we estimate the label shift among previously seen classes. Then we employ importance re-weighting of source data to formulate a single PU learning problem between source and target to estimate fraction of novel class $\widehat{w}p_t(y = k + 1)$ and to learn a discriminator f_d for the novel class. Combining discriminator and label shift corrected source classifier we get $(k+1)$-way target classifier.

D.3.1 PULSE under separability

Our ideas for PULSE framework can be extended to separability condition since (5.3) continues to hold. In particular, when OSLS satisfies the separability assumption, we may hope to jointly estimate the label shift among previously seen classes with label shift estimation techniques (Alexandari et al., 2021; Lipton et al., 2018b) and learn a domain discriminator classifier. This may be achieved by estimating label shift among examples rejected by domain discriminator classifier as belonging to previously seen classes. However, in our initial experiments, we observe that techniques proposed under strong positivity were empirically stable and outperform methods developed under separability. This is intuitive for many benchmark datasets where it may be more natural to expect that for each class there exists a subdomain that only belongs to that class than assuming separability only between novel class samples and examples from source classes.

Algorithm 16 Alternating between CVIR and BBE for Step 5 in Algorithm 8

```
input : Re-sampled training source data $\widetilde{\mathbf{X}}_1^S$, validation source data $\widetilde{\mathbf{X}}_2^S$. Training target
    data $\mathbf{X}_1^T$ and validation data $\mathbf{X}_2^T$. Hyperparameter $W, B, \delta, \gamma$.
 1: Initialize a training model $f_\theta$ and an stochastic optimization algorithm $\mathcal{A}$.
 2: $\mathbf{X}_1^N \leftarrow \mathbf{X}_1^T$.
    {// Warm start with domain discrimination training}
 3: for $i \leftarrow 1$ to $W$ do
 4:   Shuffle $(\widetilde{\mathbf{X}}_1^S, \mathbf{X}_1^N)$ into $B$ mini-batches. With $(\widetilde{\mathbf{X}}_1^S[i], \mathbf{X}_1^N[i])$ we denote $i^{\text{th}}$ mini-batch.
 5:   for $i \leftarrow 1$ to $B$ do
 6:     Set the gradient $\nabla_\theta \left[\widehat{w}\mathcal{L}^+(f_\theta; \widetilde{\mathbf{X}}_1^S[i]) + \widehat{w}\mathcal{L}^-(f_\theta; \mathbf{X}_1^N[i])\right]$ and update $\theta$ with algo-
        rithm $\mathcal{A}$.
 7:   end for
 8: end for
 9: $\widehat{w}\alpha \leftarrow \text{BBE}(\widetilde{\mathbf{X}}_2^S, \mathbf{X}_2^T, f_\theta)$                                {Algorithm 17}
10: Rank samples $x \in \mathbf{X}_1^T$ according to their loss values $\ell(f_\theta(x), -1)$.
11: $\mathbf{X}_1^N \leftarrow \{\mathbf{X}_1^T\}_{1-\widehat{w}\alpha}$ where $\{\mathbf{X}_1^T\}_{1-\widehat{w}\alpha}$ denote the lowest ranked $1 - \widehat{w}\alpha$ fraction of samples.
12: while training error $\widehat{w}\mathcal{E}^+(f_\theta; \widetilde{\mathbf{X}}_2^S) + \widehat{w}\mathcal{E}^-(f_\theta; \mathbf{X}_1^N)$ is not converged do
13:   Train model $f_\theta$ for one epoch on $(\widetilde{\mathbf{X}}_1^S, \mathbf{X}_1^N)$ as in Lines 4-7.
14:   $\widehat{w}\alpha \leftarrow \text{BBE}(\widetilde{\mathbf{X}}_2^S, \mathbf{X}_2^T, f_\theta)$                              {Algorithm 17}
15:   Rank samples $x \in \mathbf{X}_1^T$ according to their loss values $\ell(f_\theta(x), -1)$.
16:   $\mathbf{X}_1^N \leftarrow \{\mathbf{X}_1^T\}_{1-\widehat{w}\alpha}$ where $\{\mathbf{X}_1^T\}_{1-\widehat{w}\alpha}$ denote the lowest ranked $1 - \widehat{w}\alpha$ fraction of
      samples.
17: end while
output : Trained discriminator $f_d \leftarrow f_\theta$ and novel class fraction $\widehat{w}p_t(y = k+1) \leftarrow 1 - \widehat{w}\alpha$.
```

D.4 Proofs for analysis of OSLS framework

In this section, we provide missing formal statements and proofs for theorems in Sec. 5.8. This mainly includes analysing key steps of our PULSE procedure for target label marginal estimation (Step 3, 5 Algorithm 8) and learning the domain discriminator classifier (Step 5, Algorithm 8).

D.4.1 Formal statement and proof of Theorem 1

Before introducing the formal statement, we introduce some additional notation. Given probability density function p and a source classifier $f : \mathcal{X} \rightarrow \Delta^{k-1}$, define a function $q(z, j) = \int_{A(z,j)} p(x)dx$,

where $A(z, j) = \{x \in \mathcal{X} : [f(x)]_j \geqslant z\}$ for all $z \in [0, 1]$. Intuitively, $q(z, j)$ captures the cumulative density of points in a top bin for class j, i.e., the proportion of input domain that is assigned a value larger than z by the function f at the index j in the transformed space. We define an empirical estimator $\widehat{w}q(z, j)$ given a set $X = \{x_1, x_2, \ldots, x_n\}$ sampled iid from $p(x)$. Let $Z = [f(X)]_j$. Define $\widehat{w}q(z, j) = \sum_{i=1}^n \mathbb{I}\left[z_i \geqslant z\right]/n$.

Algorithm 17 Best Bin Estimation (BBE)

```
input : Re-sampled source data $\widetilde{\mathbf{X}}^S$ and target samples $\mathbf{X}^T$. Discriminator classifier
    $\widehat{f} : \mathcal{X} \to [0,1]$. Hyperparameter $0 < \delta, \gamma < 1$.
1: $Z_s, Z_t \leftarrow f(\widetilde{\mathbf{X}}^S), f(\mathbf{X}^T)$.
2: $\widehat{w}q_t(z), \widehat{w}q_s(z) \leftarrow \frac{\sum_{z_i \in Z_s} \mathbb{I}[z_i \geqslant z]}{\left|\widetilde{\mathbf{X}}^S\right|}, \frac{\sum_{z_i \in Z_t} \mathbb{I}[z_i \geqslant z]}{|\mathbf{X}|^T}$ for all $z \in [0,1]$.
3: Estimate $\widehat{w}c \leftarrow \arg\min_{c \in [0,1]} \left( \frac{\widehat{w}q_t(c)}{\widehat{w}q_s(c)} + \frac{1+\gamma}{\widehat{w}q_s(c)} \left( \sqrt{\frac{\log(4/\delta)}{2\left|\widetilde{\mathbf{X}}^S\right|}} + \sqrt{\frac{\log(4/\delta)}{2|\mathbf{X}^T|}} \right) \right)$.
output : $\widehat{w}\alpha \leftarrow \frac{\widehat{w}q_t(\widehat{w}c)}{\widehat{w}q_s(\widehat{w}c)}$
```

For each pdf p_s and p_t, we define q_s and q_t respectively. Moreover, for each class $j \in \mathcal{Y}_s$, we define $q_{t,j}$ corresponding to $p_{t,j} := p_t(x|y = j)$ and $q_{t,-j}$ corresponding to $p_{t,-j} := \frac{\sum_{i \in \mathcal{Y}_t \setminus \{j\}} p_t(y=i) p_t(x|y=i)}{\sum_{i \in \mathcal{Y}_t \setminus \{j\}} p_t(y=j)}$. Assume that we have n source examples and m target examples. Now building on BBE results from Garg et al. (2021b), we present finite sample results for target label marginal estimation:

Theorem D.4.1 (Formal statement of Theorem 5.8.1)**.** *Define* $c_j^* = \arg\min_{c \in [0,1]} \left(q_{t,-j}(c,j)/q_{t,j}(c,j)\right)$, *for all* $j \in \mathcal{Y}_s$. *Assume* $\min(n,m) \geqslant \max_{j \in \mathcal{Y}_s} \left(\frac{2\log(4k/\delta)}{q_{t,j}^2(c_j^*,j)} \right)$. *Then, for every* $\delta > 0$, $\widehat{w}p_t$ *(in Algorithm 15 with* δ *as* δ/k*) satisfies with probability at least* $1 - \delta$*, we have:*

$$\|\widehat{w}p_t - p_t\| 1 \leqslant \sum_{j \in \mathcal{Y}_s} (1 - p_t(y = j)) \left(\frac{q_{t,-j}(c_j^*, j)}{q_{t,j}(c_j^*, j)} \right) + \mathcal{O}\left(\sqrt{\frac{k^3 \log(4k/\delta)}{n}} + \sqrt{\frac{k^2 \log(4k/\delta)}{m}} \right).$$

When the data satisfies strong positivity, we observe that source classifiers often exhibit a threshold c_y on softmax output of each class $y \in \mathcal{Y}_s$ above which the *top bin* (i.e., $[c_y, 1]$) contains mostly examples from that class y. Formally, as long as there exist a threshold $c_j^* \in (0,1)$ such that $q_{t,j}(c_j^*) \geqslant \epsilon$ and $q_{t,-j}(c_j^*) = 0$ for some constant $\epsilon > 0$ for all $j \in \mathcal{Y}_s$, we show that our estimator $\widehat{w}\alpha$ converges to the true α with convergence rate $\min(n,m)^{-1/2}$. The proof technique simply builds on the proof of Theorem 1 in Garg et al. (2021b). First, we state Lemma 1 from Garg et al. (2021b). Next, for completeness we provide the proof for Theorem D.4.1 which extends proof of Theorem 1 (Garg et al., 2021b) for k classes.

Lemma D.4.2. *Assume two distributions* q_p *and* q_u *with their empirical estimators denoted by* $\widehat{w}q_p$ *and* $\widehat{w}q_u$ *respectively. Then for every* $\delta > 0$*, with probability at least* $1 - \delta$*, we have for all* $c \in [0,1]$

$$\left| \frac{\widehat{w}q_u(c)}{\widehat{w}q_p(c)} - \frac{q_u(c)}{q_p(c)} \right| \leqslant \frac{1}{\widehat{w}q_p(c)} \left(\sqrt{\frac{\log(4/\delta)}{2n_u}} + \frac{q_u(c)}{q_p(c)} \sqrt{\frac{\log(4/\delta)}{2n_p}} \right).$$

Proof of Theorem D.4.1. The main idea of the proof is to use the confidence bound derived in Lemma D.4.2 at $\widehat{w}c$ and use the fact that $\widehat{w}c$ minimizes the upper confidence bound. The proof is split into two parts. First, we derive a lower bound on $\widehat{w}q_{t,j}(\widehat{w}c_j)$ for all $j \in \mathcal{Y}_s$ and next, we use the obtained lower bound to derive confidence bound on $\widehat{w}p_t(y = j)$. With

$\widehat{w}\alpha_j$, we denote $\widehat{w}p_t(y=j)$ for all $j \in \mathcal{Y}_s$. All the statements in the proof simultaneously hold with probability $1-\delta/k$. We derive the bounds for a single $j \in \mathcal{Y}_s$ and then use union bound to combine bound for all $j \in \mathcal{Y}_s$. When it is clearly from context, we denote $q_{t,j}(c,j)$ with $q_{t,j}(c)$ and $q_t(c,j)$ with $q_t(c)$. Recall,

$$\widehat{w}c_j := \operatorname*{arg\,min}_{c \in [0,1]} \frac{\widehat{w}q_t(c)}{\widehat{w}q_{t,j}(c)} + \frac{1}{\widehat{w}q_{t,j}(c)}\left(\sqrt{\frac{\log(4k/\delta)}{2m}} + (1+\gamma)\sqrt{\frac{\log(4k/\delta)}{2np_s(y=j)}}\right) \qquad \text{and} \tag{D.3}$$

$$\widehat{w}p_t(y=j) := \frac{\widehat{w}q_t(\widehat{w}c_j)}{\widehat{w}q_{t,j}(\widehat{w}c_j)}\,. \tag{D.4}$$

Moreover,

$$c_j^* := \operatorname*{arg\,min}_{c \in [0,1]} \frac{q_t(c)}{q_{t,j}(c)} \qquad \text{and} \qquad \alpha_j^* := \frac{q_t(c_j^*)}{q_{t,j}(c_j^*)}\,. \tag{D.5}$$

Part 1: We establish lower bound on $\widehat{w}q_{t,j}(\widehat{w}c_j)$. Consider $c_j' \in [0,1]$ such that $\widehat{w}q_{t,j}(c_j') = \frac{\gamma}{2+\gamma}\widehat{w}q_{t,j}(c_j^*)$. We will now show that Algorithm 15 will select $\widehat{w}c_j < c_j'$. For any $c \in [0,1]$, we have with with probability $1-\delta/k$,

$$\widehat{w}q_{t,j}(c) - \sqrt{\frac{\log(4k/\delta)}{2n \cdot p_s(y=j)}} \leqslant q_{t,j}(c) \qquad \text{and} \qquad q_t(c) - \sqrt{\frac{\log(4k/\delta)}{2m}} \leqslant \widehat{w}q_t(c)\,. \tag{D.6}$$

Since $\frac{q_t(c_j^*)}{q_{t,j}(c_j^*)} \leqslant \frac{q_t(c)}{q_{t,j}(c)}$, we have

$$\widehat{w}q_t(c) \geqslant q_{t,j}(c)\frac{q_t(c_j^*)}{q_{t,j}(c_j^*)} - \sqrt{\frac{\log(4k/\delta)}{2m}} \geqslant \left(\widehat{w}q_{t,j}(c) - \sqrt{\frac{\log(4k/\delta)}{2n \cdot p_s(y=j)}}\right)\frac{q_t(c_j^*)}{q_{t,j}(c_j^*)} - \sqrt{\frac{\log(4k/\delta)}{2m}}\,. \tag{D.7}$$

Therefore, at c we have

$$\frac{\widehat{w}q_t(c)}{\widehat{w}q_{t,j}(c)} \geqslant \alpha_j^* - \frac{1}{\widehat{w}q_{t,j}(c)}\left(\sqrt{\frac{\log(4k/\delta)}{2m}} + \frac{q_t(c_j^*)}{q_p(c_j^*)}\sqrt{\frac{\log(4k/\delta)}{2n \cdot p_s(y=j)}}\right)\,. \tag{D.8}$$

Using Lemma D.4.2 at c^*, we have

$$\frac{\widehat{w}q_t(c)}{\widehat{w}q_{t,j}(c)} \geqslant \frac{\widehat{w}q_t(c_j^*)}{\widehat{w}q_{t,j}(c_j^*)} - \left(\frac{1}{\widehat{w}q_{t,j}(c_j^*)} + \frac{1}{\widehat{w}q_{t,j}(c)}\right)\left(\sqrt{\frac{\log(4k/\delta)}{2m}} + \frac{q_t(c_j^*)}{q_{t,j}(c_j^*)}\sqrt{\frac{\log(4k/\delta)}{2n \cdot p_s(y=j)}}\right) \tag{D.9}$$

$$\geqslant \frac{\widehat{w}q_t(c_j^*)}{\widehat{w}q_{t,j}(c_j^*)} - \left(\frac{1}{\widehat{w}q_{t,j}(c_j^*)} + \frac{1}{\widehat{w}q_{t,j}(c)}\right)\left(\sqrt{\frac{\log(4k/\delta)}{2m}} + \sqrt{\frac{\log(4k/\delta)}{2n \cdot p_s(y=j)}}\right)\,, \tag{D.10}$$

where the last inequality follows from the fact that $\alpha_j^* = \frac{q_t(c_j^*)}{q_{t,j}(c_j^*)} \leqslant 1$. Furthermore, the upper confidence bound at c is lower bound as follows:

$$\frac{\widehat{w}q_t(c)}{\widehat{w}q_{t,j}(c)} + \frac{1+\gamma}{\widehat{w}q_{t,j}(c)} \left(\sqrt{\frac{\log(4l/\delta)}{2m}} + \sqrt{\frac{\log(4k/\delta)}{2n \cdot p_s(y=j)}} \right) \tag{D.11}$$

$$\geqslant \frac{\widehat{w}q_t(c_j^*)}{\widehat{w}q_{t,j}(c_j^*)} + \left(\frac{1+\gamma}{\widehat{w}q_{t,j}(c)} - \frac{1}{\widehat{w}q_{t,j}(c_j^*)} - \frac{1}{\widehat{w}q_{t,j}(c)} \right) \left(\sqrt{\frac{\log(4k/\delta)}{2m}} + \sqrt{\frac{\log(4k/\delta)}{2n \cdot p_s(y=j)}} \right) \tag{D.12}$$

$$= \frac{\widehat{w}q_t(c_j^*)}{\widehat{w}q_{t,j}(c_j^*)} + \left(\frac{\gamma}{\widehat{w}q_{t,j}(c)} - \frac{1}{\widehat{w}q_{t,j}(c_j^*)} \right) \left(\sqrt{\frac{\log(4k/\delta)}{2m}} + \sqrt{\frac{\log(4k/\delta)}{2n \cdot p_s(y=j)}} \right) \tag{D.13}$$

Using (D.13) at $c = c'$, we have the following lower bound on ucb at c':

$$\frac{\widehat{w}q_t(c')}{\widehat{w}q_{t,j}(c')} + \frac{1+\gamma}{\widehat{w}q_{t,j}(c')} \left(\sqrt{\frac{\log(4k/\delta)}{2m}} + \sqrt{\frac{\log(4k/\delta)}{2n \cdot p_s(y=j)}} \right) \tag{D.14}$$

$$\geqslant \frac{\widehat{w}q_t(c_j^*)}{\widehat{w}q_{t,j}(c_j^*)} + \frac{1+\gamma}{\widehat{w}q_{t,j}(c_j^*)} \left(\sqrt{\frac{\log(4k/\delta)}{2m}} + \sqrt{\frac{\log(4k/\delta)}{2n \cdot p_s(y=j)}} \right), \tag{D.15}$$

Moreover from (D.13), we also have that the lower bound on ucb at $c \geqslant c'$ is strictly greater than the lower bound on ucb at c'. Using definition of $\widehat{w}c$, we have

$$\frac{\widehat{w}q_t(c_j^*)}{\widehat{w}q_{t,j}(c_j^*)} + \frac{1+\gamma}{\widehat{w}q_{t,j}(c_j^*)} \left(\sqrt{\frac{\log(4k/\delta)}{2m}} + \sqrt{\frac{\log(4k/\delta)}{2n \cdot p_s(y=j)}} \right) \tag{D.16}$$

$$\geqslant \frac{\widehat{w}q_t(\widehat{w}c)}{\widehat{w}q_{t,j}(\widehat{w}c)} + \frac{1+\gamma}{\widehat{w}q_{t,j}(\widehat{w}c)} \left(\sqrt{\frac{\log(4k/\delta)}{2m}} + \sqrt{\frac{\log(4k/\delta)}{2n \cdot p_s(y=j)}} \right), \tag{D.17}$$

and hence

$$\widehat{w}c \leqslant c'. \tag{D.18}$$

Part 2: We now establish an upper and lower bound on $\widehat{w}\alpha_j$. We start with upper confidence bound on $\widehat{w}\alpha_j$. By definition of $\widehat{w}c_j$, we have

$$\frac{\widehat{w}q_t(\widehat{w}c)}{\widehat{w}q_{t,j}(\widehat{w}c)} + \frac{1+\gamma}{\widehat{w}q_{t,j}(\widehat{w}c)} \left(\sqrt{\frac{\log(4k/\delta)}{2m}} + \sqrt{\frac{\log(4k/\delta)}{2n \cdot p_s(y=j)}} \right) \tag{D.19}$$

$$\leqslant \min_{c \in [0,1]} \left[\frac{\widehat{w}q_t(c)}{\widehat{w}q_{t,j}(c)} + \frac{1+\gamma}{\widehat{w}q_{t,j}(c)} \left(\sqrt{\frac{\log(4k/\delta)}{2m}} + \sqrt{\frac{\log(4k/\delta)}{2n \cdot p_s(y=j)}} \right) \right] \tag{D.20}$$

$$\leqslant \frac{\widehat{w}q_t(c_j^*)}{\widehat{w}q_{t,j}(c_j^*)} + \frac{1+\gamma}{\widehat{w}q_{t,j}(c_j^*)}\left(\sqrt{\frac{\log(4k/\delta)}{2m}} + \sqrt{\frac{\log(4k/\delta)}{2n\cdot p_s(y=j)}}\right). \tag{D.21}$$

Using Lemma D.4.2 at c_j^*, we get

$$\begin{aligned}\frac{\widehat{w}q_t(c_j^*)}{\widehat{w}q_{t,j}(c_j^*)} &\leqslant \frac{q_t(c_j^*)}{q_{t,j}(c_j^*)} + \frac{1}{\widehat{w}q_{t,j}(c_j^*)}\left(\sqrt{\frac{\log(4k/\delta)}{2m}} + \frac{q_t(c_j^*)}{q_{t,j}(c_j^*)}\sqrt{\frac{\log(4k/\delta)}{2n\cdot p_s(y=j)}}\right) \\ &= \alpha_j^* + \frac{1}{\widehat{w}q_{t,j}(c_j^*)}\left(\sqrt{\frac{\log(4k/\delta)}{2m}} + \alpha_j^*\sqrt{\frac{\log(4k/\delta)}{2n\cdot p_s(y=j)}}\right). \end{aligned} \tag{D.22}$$

Combining (D.21) and (D.22), we get

$$\widehat{w}\alpha_j = \frac{\widehat{w}q_t(\widehat{w}c)}{\widehat{w}q_{t,j}(\widehat{w}c)} \leqslant \alpha_j^* + \frac{2+\gamma}{\widehat{w}q_{t,j}(c_j^*)}\left(\sqrt{\frac{\log(4k/\delta)}{2m}} + \sqrt{\frac{\log(4k/\delta)}{2n\cdot p_s(y=j)}}\right). \tag{D.23}$$

Using DKW inequality on $\widehat{w}q_{t,j}(c_j^*)$, we have $\widehat{w}q_{t,j}(c_j^*) \geqslant q_{t,j}(c_j^*) - \sqrt{\frac{\log(4k/\delta)}{2n\cdot p_s(y=j)}}$. Assuming $n\cdot p_s(y=j) \geqslant \frac{2\log(4k/\delta)}{q_{t,j}^2(c_j^*)}$, we get $\widehat{w}q_{t,j}(c_j^*) \leqslant q_{t,j}(c_j^*)/2$ and hence,

$$\widehat{w}\alpha_j \leqslant \alpha_j^* + \frac{4+2\gamma}{q_{t,j}(c_j^*)}\left(\sqrt{\frac{\log(4k/\delta)}{2m}} + \sqrt{\frac{\log(4k/\delta)}{2n\cdot p_s(y=j)}}\right). \tag{D.24}$$

Finally, we now derive a lower bound on $\widehat{w}\alpha_j$. From Lemma D.4.2, we have the following inequality at $\widehat{w}c$

$$\frac{q_t(\widehat{w}c)}{q_{t,j}(\widehat{w}c)} \leqslant \frac{\widehat{w}q_t(\widehat{w}c)}{\widehat{w}q_{t,j}(\widehat{w}c)} + \frac{1}{\widehat{w}q_{t,j}(\widehat{w}c)}\left(\sqrt{\frac{\log(4k/\delta)}{2m}} + \frac{q_t(\widehat{w}c)}{q_{t,j}(\widehat{w}c)}\sqrt{\frac{\log(4k/\delta)}{2n\cdot p_s(y=j)}}\right). \tag{D.25}$$

Since $\alpha_j^* \leqslant \frac{q_t(\widehat{w}c)}{q_{t,j}(\widehat{w}c)}$, we have

$$\alpha_j^* \leqslant \frac{q_t(\widehat{w}c)}{q_{t,j}(\widehat{w}c)} \leqslant \frac{\widehat{w}q_t(\widehat{w}c)}{\widehat{w}q_{t,j}(\widehat{w}c)} + \frac{1}{\widehat{w}q_{t,j}(\widehat{w}c)}\left(\sqrt{\frac{\log(4k/\delta)}{2m}} + \frac{q_t(\widehat{w}c)}{q_{t,j}(\widehat{w}c)}\sqrt{\frac{\log(4k/\delta)}{2n\cdot p_s(y=j)}}\right). \tag{D.26}$$

Using (D.24), we obtain a very loose upper bound on $\frac{\widehat{w}q_t(\widehat{w}c)}{\widehat{w}q_{t,j}(\widehat{w}c)}$. Assuming $\min(n\cdot p_s(y=j), m) \geqslant \frac{2\log(4k/\delta)}{q_{t,j}^2(c_j^*)}$, we have $\frac{\widehat{w}q_t(\widehat{w}c)}{\widehat{w}q_{t,j}(\widehat{w}c)} \leqslant \alpha_j^* + 4 + 2\gamma \leqslant 5 + 2\gamma$. Using this in (D.26), we have

$$\alpha_j^* \leqslant \frac{\widehat{w}q_t(\widehat{w}c)}{\widehat{w}q_{t,j}(\widehat{w}c)} + \frac{1}{\widehat{w}q_{t,j}(\widehat{w}c)}\left(\sqrt{\frac{\log(4k/\delta)}{2m}} + (5+2\gamma)\sqrt{\frac{\log(4k/\delta)}{2n\cdot p_s(y=j)}}\right). \tag{D.27}$$

Moreover, as $\widehat{w}c \geqslant c'$, we have $\widehat{w}q_{t,j}(\widehat{w}c) \geqslant \frac{\gamma}{2+\gamma}\widehat{w}q_{t,j}(c_j^*)$ and hence,

$$\alpha_j^* - \frac{\gamma+2}{\gamma\widehat{w}q_{t,j}(c_j^*)}\left(\sqrt{\frac{\log(4k/\delta)}{2m}} + (5+2\gamma)\sqrt{\frac{\log(4k/\delta)}{2n\cdot p_s(y=j)}}\right) \leqslant \frac{\widehat{w}q_t(\widehat{w}c)}{\widehat{w}q_{t,j}(\widehat{w}c)} = \widehat{w}\alpha_j\,. \tag{D.28}$$

As we assume $n \cdot p_s(y=j) \geqslant \frac{2\log(4k/\delta)}{q_{t,j}^2(c_j^*)}$, we have $\widehat{w}q_{t,j}(c_j^*) \leqslant q_{t,j}(c_j^*)/2$, which implies the following lower bound on α:

$$\alpha_j^* - \frac{2\gamma+4}{\gamma q_{t,j}(c_j^*)}\left(\sqrt{\frac{\log(4k/\delta)}{2m}} + (5+2\gamma)\sqrt{\frac{\log(4k/\delta)}{2n\cdot p_s(y=j)}}\right) \leqslant \widehat{w}\alpha_j\,. \tag{D.29}$$

Combining lower bound (D.29) and upper bound (D.24), we get

$$\left|\widehat{w}\alpha_j - \alpha_j^*\right| \leqslant l_j\left(\sqrt{\frac{\log(4k/\delta)}{2m}} + \sqrt{\frac{\log(4k/\delta)}{2n\cdot p_s(y=j)}}\right)\,, \tag{D.30}$$

for some constant l_j. Additionally by our assumption of OSLS problem $p_s(y=j) > c/k$ for some constant $c > 0$, we have

$$\left|\widehat{w}\alpha_j - \alpha_j^*\right| \leqslant l_j'\left(\sqrt{\frac{\log(4k/\delta)}{2m}} + \sqrt{\frac{k\log(4k/\delta)}{2n}}\right)\,, \tag{D.31}$$

for some constant l_j'.

Combining the above obtained bound for all $j \in \mathcal{Y}_s$ with union bound, we get with probability at least $1-\delta$,

$$\sum_{j\in\mathcal{Y}_s}\left|\widehat{w}\alpha_j - \alpha_j^*\right| \leqslant l_{\max}'\left(\sqrt{\frac{k^2\log(4k/\delta)}{2m}} + \sqrt{\frac{k^3\log(4k/\delta)}{2n}}\right)\,, \tag{D.32}$$

where $l_{\max}' = \max l_j'$. Now, note that for each $j \in \mathcal{Y}_s$, we have $q_t(c) = p_t(y=j)\cdot q_{t,j}(c) + (1-p_t(y=j))\cdot q_{t,-j}(c)$. Hence $\alpha_j^* = p_t(y=j) + (1-p_t(y=j))\cdot q_{t,-j}(c)/\cdot q_{t,j}(c)$. Plugging this in, we get the desired bound. □

Intuitively, the guarantees in the previous theorem capture the tradeoff due to the proportion of negative examples in the top bin (bias) versus the proportion of positives in the top bin (variance). As a corollary, we can show convergence to true mixture if there exits c_j^* for all $j \in \mathcal{Y}_s$ such that $q_{t,-j}(c_j^*, j) = 0$ and $q_{t,j}(c_j^*, j) \geqslant \epsilon$ for some $\epsilon > 0$. Put simply, efficacy of BBE relies on existence of a threshold on probability scores assigned by the classifier such that the examples mapped to a score greater than the threshold are *mostly* positive. Using the terminology from Garg et al. (2021b), we refer to this as the top bin property. Next, we provide empirical evidence of this property while using the source classifier to estimate the relative proportion of target label marginal among source classes.

Empirical evidence of the top bin property We now empirically validate the positive pure top bin property (Fig. D.1). We include results with Resnet-18 trained on the CIFAR10 OSLS setup same as our main experiments. We observe that source classifier approximately satisfies the positive pure top bin property for small enough top bin sizes.

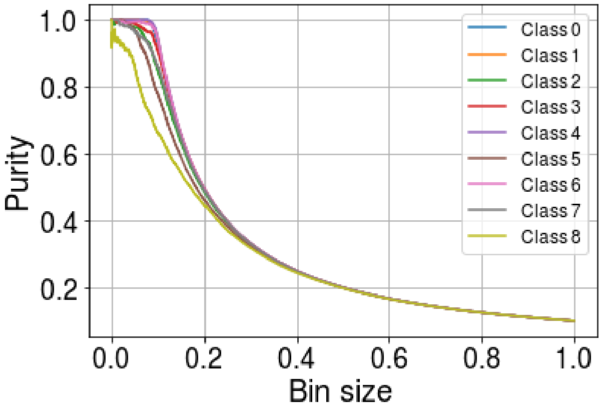


Figure D.1: Purity and size (in terms of fraction of unlabeled samples) in the top bin for all classes. Bin size refers to the fraction of examples in the top bin. With purity, we refer to the fraction of examples from a specific class j in the top bin. Results with ResNet-18 on CIFAR10 OSLS setup. Details of the setup in App. D.6.2. As the bin size increases for all classes the purity decreases.

D.4.2 Formal statement and proof of Theorem 2

In this section, we show that in population on a separable Gaussian dataset, CVIR will recover the optimal classifier. Note that here we consider a binary classification problem similar to the one in Step 5 in Algorithm 8. Since we are primarily interested in analysing the iterative procedure for obtaining domain discriminator classifier, we assume that α is known.

In population, we have access to positive distribution (i.e., p_p), unlabeled distribution (i.e., $p_u := \alpha p_p + (1-\alpha)p_n$), and mixture coefficient α. Our goal is to recover the classifier that discriminates p_p versus p_n.

For ease, we re-introduce some notation. For a classifier f and loss function ℓ, define

$$\text{VIR}_\alpha(f) = \inf\{\tau \in \mathbb{R} : \text{P}_{x\sim p_u}(\ell(x,-1;f) \leqslant \tau) \geqslant 1-\alpha\}\,. \tag{D.33}$$

Intuitively, $\text{VIR}_\alpha(f)$ identifies a threshold τ to capture bottom $1-\alpha$ fraction of the loss $\ell(x,-1)$ for points x sampled from p_u. Additionally, define CVIR loss as

$$\mathcal{L}(f,w) = \alpha\mathbb{E}_{p_p}\left[\ell(x,1;f)\right] + \mathbb{E}_{p_u}\left[w(x)\ell(x,-1;f)\right]\,, \tag{D.34}$$

for classifier f and some weights $w(x) \in \{0,1\}$. Recall that given a classifier f_t at an iterate t, CVIR procedure proceeds as follows:

$$w_t(x) = \mathbb{I}\left[\ell(x,-1;f_t) \leqslant \text{VIR}_\alpha(f_t)\right]\,, \tag{D.35}$$

$$f_{t+1} = f_t - \eta\nabla\mathcal{L}_f(f_t,w_t)\,. \tag{D.36}$$

We assume a data generating setup with where the support of positive and negative data is completely disjoint. We assume that x are drawn from two half multivariate Gaussian with mean zero and identity covariance, i.e.,

$$\begin{aligned} x \sim p_p \Leftrightarrow x &= \gamma_0 \theta_{\text{opt}} + z |\, \theta_{\text{opt}}^T z \geqslant 0, \text{ where } z \sim \mathcal{N}(0, I_d) \\ x \sim p_n \Leftrightarrow x &= -\gamma_0 \theta_{\text{opt}} + z |\, \theta_{\text{opt}}^T z < 0, \text{ where } z \sim \mathcal{N}(0, I_d) \end{aligned}$$

Here γ_0 is the margin and $\theta_{\text{opt}} \in \mathbb{R}^d$ is the true separator. Here, we have access to distribution p_p and $p_u = \alpha p_p + (1-\alpha) p_n$. Assume ℓ as the logistic loss. For simplicity, we will denote $\mathcal{L}(f_{\theta_t}, w_t)$ with $\mathcal{L}(\theta_t, w_t)$.

Theorem D.4.3 (Formal statement of Theorem 5.8.2)**.** *In the data setup described above, a linear classifier $f(x;\theta) = \sigma\left(\theta^T x\right)$ initialized at some θ_0 such that $\mathcal{L}(\theta_0, w_0) < \log(2)$, trained with CVIR procedure as in equations* (D.35)-(D.36) *will converge to an optimal positive versus negative classifier.*

Proof of Theorem D.4.3. The proof uses two key ideas. One, at convergence of the CVIR procedure, the gradient of CVIR loss in (D.34) converges to zero. Second, for any classifier θ that is not optimal for positive versus negative classification, we show that the CVIR gradient in (D.34) is non-zero.

Part 1 We first show that the loss function $\mathcal{L}(\theta, w)$ in (D.34) is 2-smooth with respect to θ for fixed w. Using gradient descent lemma with the decreasing property of loss in (D.35)-(D.36), we show that gradient converges to zero eventually. Considering gradient of $\mathcal{L}$, we have

$$\nabla_\theta \mathcal{L}(\theta, w) = \alpha \mathbb{E}_{p_p}\left[(f(x;\theta) - 1)x\right] + \mathbb{E}_{p_u}\left[w(x)(f(x;\theta) - 0)x\right] . \tag{D.37}$$

Moreover, $\nabla^2 \mathcal{L}$ is given by

$$\nabla_\theta^2 \mathcal{L}(\theta, w) = \alpha \mathbb{E}_{p_p}\left[\nabla f(x;\theta) x x^T\right] + \mathbb{E}_{p_u}\left[w(x) \nabla f(x;\theta) x x^T\right] . \tag{D.38}$$

Since $\nabla f(x;\theta) \leqslant 1$, we have $v^T \nabla^2 \mathcal{L} v \leqslant 2$ for all unit vector $v \in R^d$. Now, by gradient descent lemma if $\eta \leqslant 1/2$, at any step t we have, $\mathcal{L}(\theta_{t+1}, w_t) \leqslant \mathcal{L}(\theta_t, w_t)$. Moreover, by definition of $\text{VIR}_\alpha(\theta)$ in (D.33) and update (D.35), we have $\mathcal{L}(\theta_{t+1}, w_{t+1}) \leqslant \mathcal{L}(\theta_{t+1}, w_t)$. Hence, we have $\mathcal{L}(\theta_{t+1}, w_{t+1}) \leqslant \mathcal{L}(\theta_t, w_t)$. Since, the loss is lower bounded from below at 0, for every $\epsilon > 0$, we have for large enough t (depending on ϵ), $\|\nabla_\theta \mathcal{L}(\theta_t, w_t)\| 2 \leqslant \epsilon$, i.e., $\|\nabla_\theta \mathcal{L}(\theta_t, w_t)\| 2 \to 0$ as $t \to \infty$.

Part 2 Consider a general scenario when $\gamma > 0$. Denote the input domain of p_p and p_n as P and N respectively. At any step t, for all points $x \in \mathcal{X}$ such that $p_u(x) > 0$ and $w_t(x) = 0$, we say that x is rejected from p_u. We denote the incorrectly rejected subdomain of p_n from p_u as N_r and the incorrectly accepted subdomain of p_p from p_u as P_a. Formally, $N_r = \{x : p_n(x) > 0 \text{ and } w_t(x) = 0\}$ and $P_a = \{x : p_p(x) > 0 \text{ and } w_t(x) = 1\}$. We will show that $p_p(P_a) \to 0$ as $t \to \infty$, and hence, we will recover the optimal classifier where we reject none of p_u incorrectly.

Observe that at any time t, for fixed w_t and $\theta = \theta_t$, the gradient of CVIR loss in (D.34), can be expressed as:

$$\nabla_\theta \mathcal{L}(\theta, w_t) = \underbrace{\alpha \int_{x \in P \setminus P_a} (f(x;\theta) - 1) x \cdot p_p(x) dx}_{\text{I}} + \underbrace{(1-\alpha) \int_{x \in N \setminus N_r} (f(x;\theta) - 0) x \cdot p_n(x) dx}_{\text{II}}$$
$$+ \underbrace{\alpha \int_{x \in P_a} (2f(x;\theta) - 1) x \cdot p_p(x) dx}_{\text{III}} \,. \tag{D.39}$$

Note that for any x, θ, $0 \leqslant f(x;\theta) \leqslant 1$. Now consider inner product of individual terms above with θ_{opt}, we get

$$\langle \text{I}, \theta_{\text{opt}} \rangle = \int_{x \in P \setminus P_a} (f(x;\theta) - 1) x^T \theta_{\text{opt}} \cdot p_p(x) dx \leqslant -\gamma_0 \int_{x \in P \setminus P_a} (1 - f(x;\theta)) \cdot p_p(x) dx \,, \tag{D.40}$$

$$\langle \text{II}, \theta_{\text{opt}} \rangle = \int_{x \in N \setminus N_r} (f(x;\theta) - 0) x^T \theta_{\text{opt}} \cdot p_n(x) dx \leqslant -\gamma_0 \int_{x \in N \setminus N_r} (f(x;\theta) - 0) \cdot p_n(x) dx \,, \tag{D.41}$$

$$\langle \text{III}, \theta_{\text{opt}} \rangle = \int_{x \in P_a} (2f(x;\theta) - 1) x^T \theta_{\text{opt}} \cdot p_p(x) dx \leqslant -\gamma_0 \int_{x \in P_a} (1 - 2f(x;\theta)) \cdot p_p(x) dx \,. \tag{D.42}$$

Now, we will argue that individually all the three LHS terms in (D.40), (D.41), (D.42) are negative for all classifiers that do not separate positive versus negative data begining from $\mathcal{L}(\theta_0, w_0) < \log(2)$. And hence, we show that these terms approach zero individually only when the linear classifier approaches an optimal positive versus negative classifier.

First, we consider the term in the LHS of equation (D.42). When $\alpha = 0.5$, we have $\text{VIR}_\alpha(\theta) = 0.5$ and hence, $(1 - 2f(x;\theta)) \leqslant 0$ for $x \in P_a$. When $\alpha > 0.5$, $\text{VIR}_\alpha(\theta) < 0.5$ because, the proportion $\alpha \cdot p_p(P_a)$ matches with proportion $(1-\alpha) \cdot p_n(N_r)$. Hence, we again have $(1 - 2f(x;\theta)) \leqslant 0$ for $x \in P_a$.

To handle the case with $\alpha < 0.5$, we use a symmetry of he distribution to because $\text{VIR}_\alpha(\theta) > 0.5$ and $(1 - 2f(x;\theta))$ can take positive and negative values. However, note that $\text{VIR}_\alpha(\theta)$ will be selected such that the proportion $\alpha \cdot p_p(P_a)$ matches with proportion $(1-\alpha) \cdot P_n(N_r)$. In particular, we can split P_a into three disjoint sets $P_a^{(1)}$, $P_a^{(2)}$, and $P_a^{(3)}$ such that for all $x \in P_a^{(1)}$ we have $f(x;\theta) >= 0.5$, for all $x \in P_a^{(2)} \cup P_a^{(3)}$ we have $f(x;\theta) < 0.5$ and $p_p(P_a^{(3)}) = \frac{\alpha}{1-\alpha} p_p(N_r)$. Additionally, by symmetry of distribution around θ, we have $\int_{x \in P_a^{(1)}} (1 - 2f(x;\theta)) \cdot p_p(x) dx + \int_{x \in P_a^{(2)}} (1 - 2f(x;\theta)) \cdot p_p(x) dx = 0$. Hence, we get

$$\langle \text{III}, \theta_{\text{opt}} \rangle \leqslant -\gamma_0 \int_{x \in P_a} (1 - 2f(x;\theta)) \cdot p_p(x) dx = -\gamma_0 \int_{x \in P_a^{(3)}} (1 - 2f(x;\theta)) \cdot p_p(x) dx \,. \tag{D.43}$$

Combining all three cases, we get $\langle \text{III}, \theta_{\text{opt}} \rangle < 0$ when $p_p(P_a) > 0$.

Now we consider LHS terms in (D.40) and (D.41). Note that for all $x \in P \cup N$, we have $0 \leqslant f(x) \leqslant 1$. Thus with $p_p(P \backslash P_a) > 0$, $\langle \text{I}, \theta_{\text{opt}} \rangle \to 0$ when $f(x, \theta) \to 1$ for all $x \in P \backslash P_a$. Similarly with $p_n(N \backslash N_r) > 0$, $\langle \text{II}, \theta_{\text{opt}} \rangle \to 0$ when $f(x, \theta) \to 0$ for all $x \in N \backslash N_r$.

From part 1, for gradient $\|\nabla_\theta \mathcal{L}(\theta_t, w_t)\| 2$ to converge to zero as $t \to \infty$, we must have that LHS in equations (D.40), (D.41), and (D.42) converges to zero individually. Since CVIR loss decreases continuously and $\mathcal{L}(\theta_0, w_0) < \log(2)$, we have that $p_p(P_a) \to 0$ and hence, $f(x, \theta) \to 1$ for all $x \in P$ and $f(x, \theta) \to 0$ for all $x \in N$.

□

The above analysis can be extended to show convergence to max-margin classifier by using arguments from Soudry et al. (2018). In particular, as $p_p(P_a) \to 0$, we can show that $\theta_t / \|\theta_t\| 2$ will converge to the max-margin classifier for p_p versus p_n, i.e., θ_{opt} if $p_p(P_a) \to 0$ in finite number of steps. Note that we need an assumption that the initialized model θ_0 is strictly better than a model that randomly guesses or initialized at all zeros. This is to avoid convergence to the local minima of $\theta = \mathbf{0}$ with CVIR training. This assumption is satisfied when the classifier is initialized in a way such that $\langle \theta_0, \theta_{\text{opt}} \rangle > 0$. In general, we need a weaker assumption that during training with any randomly initialized classifier, there exists an iterate t during CVIR training such that $\langle \theta_t, \theta_{\text{opt}} \rangle > 0$.

D.4.3 Extension of Theorem 1

We also extend the analysis in the proof of Theorem D.4.1 to Step 5 of Algorithm 8 to show convergence of estimate $\widehat{w}p_t(y = k+1)$ to true prevalence $p_t(y = k+1)$. In particular, we show that the estimation error for prevalence of the novel class will primarily depend on sum of two terms: (i) error in approximating the label shift corrected source distribution, i.e., $p'_s(x)$; and (ii) purity of the top bin of the domain discriminator classifier.

Before formally introducing the result, we introduce some notation. Similar to before, given probability density function p and a domain discriminator classifier $f : \mathcal{X} \to \Delta$, define a function $q = \int_{A(z)} p(x) dx$, where $A(z) = \{x \in \mathcal{X} : f(x) \geqslant z\}$ for all $z \in [0, 1]$. Intuitively, $q(z)$ captures the cumulative density of points in a top bin, i.e., the proportion of input domain that is assigned a value larger than z by the function f in the transformed space. We denote $p_t(x | y = k+1)$ with $p_{t,k+1}$. For each pdf p_t, $p_{t,k+1}$, and p'_s, we define q_t, $q_{t,k+1}$, and q'_s respectively. Note that since We define an empirical estimator $\widehat{w}q(z)$ given a set $X = \{x_1, x_2, \ldots, x_n\}$ sampled iid from $p(x)$. Let $Z = f(X)$. Define $\widehat{w}q(z) = \sum_{i=1}^{n} \mathbb{I}\left[z_i \geqslant z\right] / n$.

Recall that in Step 5 of Algorithm 8, to estimate the proportion of novel class, we have access to re-sampled data from approximate label shift corrected source distribution $\widehat{w}q'_s(x)$. Assume that we the size of re-sampled dataset is n.

Theorem D.4.4. *Define* $c^* = \arg\min_{c \in [0,1]} \left(q_{t,k+1}(c) / \widehat{w}q'_s(c) \right)$. *Assume* $\min(n, m) \geqslant \left(\frac{2 \log(4/\delta)}{(\widehat{w}q'_s(c^*))^2} \right)$. *Then, for every* $\delta > 0$, $[\widehat{w}p_t]_{k+1} := \widehat{w}p_t(y = k+1)$ *in Step 5 of Algorithm 8* *satisfies with probability at least* $1-\delta$, *we have:*

$$\left|[\widehat{w}p_t]_{k+1} - [p_t]_{k+1}\right| \leqslant (1-[p_t]_{k+1}) \underbrace{\frac{|q_s'(c^*) - \widehat{w}q_s'(c^*)|}{\widehat{w}q_s'(c^*)}}_{\substack{\textit{Error in estimating}\\ \textit{label shift corrected source}}} + [p_t]_{k+1} \underbrace{\left(\frac{q_{t,k+1}(c^*)}{\widehat{w}q_s'(c^*)}\right)}_{\substack{\textit{Impurity in}\\ \textit{top bin}}}$$

$$+\,\mathcal{O}\left(\sqrt{\frac{\log(4/\delta)}{n}} + \sqrt{\frac{\log(4/\delta)}{m}}\right).$$

Proof. We can simply prove this theorem as Corollary of Theorem 1 from Garg et al. (2021b). Note that $q_t(c^*) = (1-p_t(y=k+1)) \cdot q_s'(c^*) + p_t(y=k+1) \cdot q_{t,k+1}(c^*)$. Adding and subtracting $(1-p_t(y=k+1)) \cdot \widehat{w}q_s'(c^*)$ and dividing by $\widehat{w}q_s'$, we get $\frac{q_t(c^*)}{\widehat{w}q_s'(c^*)} = (1-p_t(y=k+1)) \cdot \frac{|q_s'(c^*)-\widehat{w}q_s'(c^*)|}{\widehat{w}q_s'(c^*)} + (1-p_t(y=k+1)) + p_t(y=k+1) \cdot \frac{q_{t,k+1}(c^*)}{\widehat{w}q_s'(c^*)}$. Plugging in bound for LHS from Theorem 1 in Garg et al. (2021b), we get the desired result. □

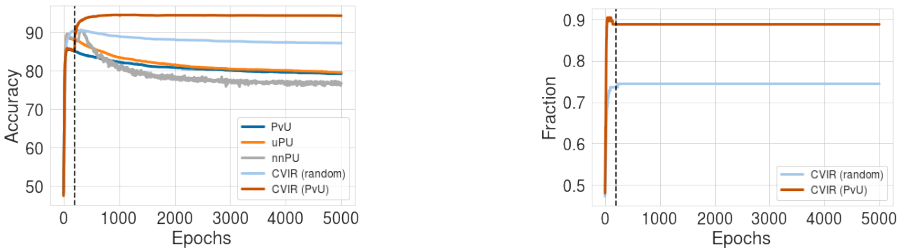
D.4.4 Extensions of Theorem 2 to general separable datasets

For general separable datasets, CVIR has undesirable property of getting stuck at local optima where gradient in (D.42) can be zero by maximizing entropy on the subset P_a which is (incorrectly) not-rejected from p_u in CVIR iterations. Intuitively, if the classifier can perfectly separate $P \backslash P_a$ and $N \backslash N_r$ and at the same time maximize the entropy of the region P_a, then the classifier trained with CVIR can get stuck in this local minima.

However, we can extend the above analysis with some modifications to the CVIR procedure. Note that when the CVIR classifier maximizes the entropy on P_a. it makes an error on points in P_a. Since, we have access to the distribution p_p, we can add an additional regularization penalty to the CVIR loss that ensures that the converged classifier with CVIR correctly classifies all the points in p_p. With a large enough regularization constant for the supervised loss on p_p, we can dominate the gradient term in (D.42) which pushes CVIR classifier to correct decision boundary even on P_a (instead of maximizing entropy). We leave formal analysis of this conjecture for future work. Since we warm start CVIR training with a positive versus unlabeled classifier, if we obtain an initialization close enough to the true positive versus negative decision boundary, by monotonicity property of CVIR iterations, we may never get stuck in such a local minima even without modifications to loss.

D.5 Empirical investigation of CVIR in toy setup

As noted in our ablation experiments and in Garg et al. (2021b), domain discriminator trained with CVIR outperforms classifiers trained with other consistent objectives (nnPU (Kiryo et al., 2017) and uPU (Du Plessis et al., 2015)). While the analysis in Sec. 5.8 highlights consistency of CVIR procedure in population, it doesn't capture the observed empirical efficacy of CVIR over alternative methods in overparameterized models. In the Gaussian setup described in Sec. D.4.2, we train overparameterized linear models to



(a) Accuracy on validation positive versus negative data

(b) Fraction of correctly rejected examples with CVIR

Figure D.2: **Comparison of different methods in overparameterized toy setup.** CVIR (random) denotes CVIR with random initialization and CVIR (PvU) denotes warm start with a positive versus negative classifier. Vertical line denotes the epoch at which we switch from PvU to CVIR in CVIR (PvU) training. (a) We observe that CVIR (PvU) improves significantly even over the best early stopped PvU model. As training proceeds, we observe that accuracy of nnPU, uPU and PvU training drops whereas CVIR (random) and CVIR (PvU) maintains superior and stable performance. (b) We observe that warm start training helps CVIR over randomly initialized model to correctly identity positives among unlabeled for rejection.

compare CVIR with other methods (Fig. D.2). We fix $d = 1000$ and use $n = 250$ positive and $m = 250$ unlabeled points for training with $\alpha = 0.5$. We set the margin γ at 0.05. We compare CVIR with unbiased losses uPU and nnPU. We also make comparison with a naive positive versus unlabeled classifier (referred to as PvU). For CVIR, we experiment with a randomly initialized classifier and initialized with a PvU classifier trained for 200 epochs.

First, we observe that when a classifier is trained to distinguish positive and unlabeled data, *early learning* happens (Arora et al., 2019a; Garg et al., 2021a; Liu et al., 2020), i.e., during the initial phase of learning classifier learns to classify positives in unlabeled correctly as positives achieving high accuracy on validation positive versus negative data. While the early learning happens with all methods, soon in the later phases of training PvU starts overfitting to the unlabeled data as negative hurting its validation performance. For uPU and nnPU, while they improve over PvU training during the initial epochs, the loss soon becomes biased hurting the performance of classifiers trained with uPU and nnPU on validation data.

For CVIR trained from a randomly initialized classifier, we observe that it improves slightly over the best PvU or the best nnPU model. Moreover, it maintains a relatively stable performance throughout the training. CVIR initialized with a PvU classifier significantly improves the performance. In Fig. D.2 (b), we show that CVIR initialized with a PvU correctly rejects significantly more fraction of positives from unlabeled than CVIR trained from scratch. Thus, post early learning rejection of large fraction of positives from unlabeled training in equation (5.4) crucially helps CVIR.

D.6 Experimental Details

D.6.1 Baselines

We compare PULSE with several popular methods from OSDA literature. While these methods are not specifically proposed for OSLS, they are introduced for the more general OSDA problem. In particular, we make comparions with DANCE (Saito et al., 2020), UAN (You et al., 2019), CMU (Fu et al., 2020), STA (Liu et al., 2019a), Backprop-ODA (or BODA) (Saito et al., 2018b). We use the open source implementation available at `https://github.com/thuml` and `https://github.com/VisionLearningGroup/DANCE/`. Since OSDA methods do not estimate the prevalence of novel class explicitly, we use the fraction of examples predicted in class $k+1$ as a surrogate. We next briefly describe the main idea for each method:

Backprob-ODA Saito et al. (2018b) proposed backprob ODA to train a $(k+1)$-way classifier. In particular, the network is trained to correctly classify source samples and for target samples, the classifier (specifically the last layer) is trained to output 0.5 for the probability of the unknown class. The feature extractor is trained adversarially to move the probability of unknown class away from 0.5 on target examples by utilizing the gradient reversal layer.

Separate-To-Adapt (STA) Liu et al. (2019a) trained a network that learns jointly from source and target by learning to separate negative (novel) examples from target. The training is divided into two parts. The first part consists of training a multi-binary $G_c|_{c=1}^{|\mathcal{Y}_s|}$ classifier on labeled source data for each class and a binary classifier G_b which generates the weights w for rejecting target samples in the novel class. The second part consists of feature extractor G_f, a classifier G_y and domain discriminator G_d to perform adversarial domain adaptation between source and target data in the source label space. G_y and G_d are trained with incorporating weights w predicted by G_b in the first stage.

Calibrated Multiple Uncertainties (CMU) Fu et al. (2020) trained a source classifier and a domain discriminator to discriminate the novel class from previously seen classes in target. To train the discriminator network, CMU uses a weighted binary cross entropy loss where $w(x)$ for each example x in target which is the average of uncertainty estimates, e.g. prediction confidence of source classifier. During test time, target data x with $w(x) \geqslant w_0$ (for some pre-defined threshold w_0) is classified as an example from previously seen classes and is given a class prediction with source classifier. Otherwise, the target example is classified as belonging to the novel class.

DANCE Saito et al. (2020) proposed DANCE which combines a self-supervised clustering loss to cluster neighboring target examples and an entropy separation loss to consider alignment with source. Similar to CMU, during test time, DANCE uses thresholded prediction entropy of the source classifier to classifier a target example as belonging to the novel class.

Universal Adaptation Networks (UAN) You et al. (2019) proposed UAN which also trains a source classifier and a domain discriminator to discriminate the novel class from previously seen classes in target. The objective is similar to CMU where instead of using uncertainty estimates from multiple classifiers, UAN uses prediction confidence of domain discriminator classifier. Similar to CMU, at test time, target data x with $w(x) \leqslant w_0$ (for some pre-defined threshold w_0) is classified as an example from previously seen classes and is given a class prediction with source classifier. Otherwise, the target example is classified as belonging to the novel class.

For alternative baselines, we experiment with source classifier directly deployed on the target data which may contain novel class and label shift among source classes (referred to as *source-only*). This naive comparison is included to quantify benefits of label shift correction and identifying novel class over a typical k-way classifiers.

We also train a domain discriminator classifier for source versus target (referred to as *domain disc.*). This is an adaptation of PU learning baseline(Elkan and Noto, 2008) which assumes no label shift among source classes. We use simple domain discriminator training to distinguish source versus target. To estimate the fraction of novel examples, we use the EN estimator proposed in Elkan and Noto (2008). For any target input, we make a prediction with the domain discriminator classifier (after re-scaling the sigmoid output with the estimate proportion of novel examples). Any example that is classified as target, we assign it the class $k + 1$. For examples classified as source, we make a prediction for them using the k-way source classifier.

Finally, per the reduction presented in Sec. 5.5, we train k PU classifiers (referred to as *k-PU*). To train each PU learning classifier, we can plugin any method discussed in Sec. C.5. In the main paper, we included results obtained with plugin state-of-the-art PU learning algorithms. In App. D.6.8, we present ablations with other PU learning methods.

D.6.2 Dataset and OSLS Setup Details

We conduct experiments with seven benchmark classification datasets across vision, natural language, biology and medicine. Our datasets span language, image and table modalities. For each dataset, we simulate an OSLS problem. We experiment with different fraction of novel class prevalence, source label distribution, and target label distribution. We randomly choose classes that constitute the novel target class. After randomly choosing source and novel classes, we first split the training data from each source class randomly into two partitions. This creates a random label distribution for shared classes among source and target. We then club novel classes to assign them a new class (i.e. $k+1$). Finally, we throw away labels for the target data to obtain an unsupervised DA problem. We repeat the same process on iid hold out data to obtain validation data with no target labels. For main experiments in the paper, we next describe important details for the OSLS setup simulated. All the other details can be found in the code repository.

For vision, we use CIFAR10, CIFAR100 (Krizhevsky and Hinton, 2009) and Entity30 (Santurkar et al., 2021). For language, we experiment with Newsgroups-20 dataset. Additionally, inspired by applications of OSLS in biology and medicine, we experiment with Tabula

Muris (Consortium et al., 2020) (Gene Ontology prediction), Dermnet (skin disease prediction), and BreakHis (Spanhol et al., 2015) (tumor cell classification).

CIFAR10 For CIFAR10, we randomly select 9 classes as the source classes and a novel class formed by the remaining class. After randomly sampling the label marginal for source and target randomly, we get the prevalence for novel class as 0.2152.

CIFAR100 For CIFAR100, we randomly select 85 classes as the source classes and a novel class formed by aggregating the data from 15 remaining classes. After randomly sampling the label marginal for source and target randomly, we get the prevalence for novel class as 0.2976.

Entity30 Entity30 is a subset of ImageNet (Russakovsky et al., 2015) with 30 super classes. For Entity30, we randomly select 24 classes as the source classes and a novel class formed by aggregating the data from 6 remaining classes. After randomly sampling the label marginal for source and target randomly, we get the prevalence for novel class as 0.3942.

Newgroups-20 For Newsgroups20[1], we randomly select 16 classes as the source classes and a novel class formed by aggregating the data from 4 remaining classes. After randomly sampling the label marginal for source and target randomly, we get the prevalence for novel class as 0.3733. This dataset is motivated by scenarios where novel news categories can appear over time but the distribution of articles given a news category might stay relatively unchanged.

BreakHis BreakHis[2] contains 8 categories of cell types, 4 types of benign breast tumor and 4 types malignant tumors (breast cancer). Here, we simulate OSLS problem specifically where 6 cell types are observed in the source (3 from each) and a novel class appears in the target with 1 cell type from each category. After randomly sampling the label marginal for source and target randomly, we get the prevalence for novel class as 0.2708.

Dermnet Dermnet data contains images of 23 types of skin diseases taken from Dermnet NZ[3]. We simulate OSLS problem specifically where 18 diseases are observed in the source and a novel class appears in the target with the rest of the 5 diseases. After randomly sampling the label marginal for source and target randomly, we get the prevalence for novel class as 0.3133.

Tabula Muris Tabula Muris dataset (Consortium et al., 2020) comprises of different cell types collected across 23 organs of the mouse model organism. We use the data preprocessing scripts provided in (Cao et al., 2021)[4]. We just use the training set comprising of 57 classes for our experiments. We simulate OSLS problem specifically where 28 cell types are observed in the source and a novel class appears in the target with the rest of the 29 cell types. After randomly sampling the label marginal for source and target randomly,

[1] http://qwone.com/~jason/20Newsgroups/

[2] https://web.inf.ufpr.br/vri/databases/breast-cancer-histopathological-database-breakhis/

[3] http://www.dermnet.com/dermatology-pictures-skin-disease-pictures

[4] https://github.com/snap-stanford/comet

we get the prevalence for novel class as 0.6366.

D.6.3 Details on the Experimental Setup

We use Resnet18 (He et al., 2016) for CIFAR10, CIFAR100, and Entity30. For all three datasets, in our main experiments, we train Resnet-18 from scratch. We use SGD training with momentum of 0.9 for 200 epochs. We start with learning rate 0.1 and decay it by multiplying it with 0.1 every 70 epochs. We use a weight decay of 5×10^{-4}. For CIFAR100 and CIFAR10, we use batch size of 200. For Entity30, we use a batch size of 32. In App. D.6.7, we experiment with contrastive pre-training instead of random initialization.

For newsgroups, we use a convolutional architecture[5]. We use glove embeddings to initialize the embedding layer. We use Adam optimizer with a learning rate of 0.0001 and no weight decay. We use a batch size of 200. We train with constant learning rate for 120 epochs.

For Tabular Muris, we use the fully connected MLP used in Cao et al. (2021). We use the hyperparameters used in Cao et al. (2021). We use Adam optimizer with a learning rate of 0.0001 and no weight decay. We train with constant learning rate for 40 epochs. We use a batch size of 200.

For Dermnet and BreakHis, we use Resnet-50 pre-trained on Imagenet. We use an initial learning rate of 0.0001 and decay it by 0.96 every epoch. We use SGD training with momentum of 0.9 and weight decay of 5×10^{-4}. We use a batch size of 32. These are the default hyperparameters used in Alom et al. (2019) and Liao (2016).

For all methods, we use the same backbone for discriminator and source classifier. Additionally, for PULSE and domain disc., we use the exact same set of hyperparameters to train the domain discriminator and source classifier. For kPU, we use a separate final layer for each class with the same backbone. We use the same hyperparameters described above for all three methods. For OSDA methods, we use default method specific hyperparameters introduced in their works. Since we do not have access to labels from the target data, we do not perform hyperparameter tuning but instead use the standard hyperparameters used for training on labeled source data. In future, we may hope to leverage heuristics proposed for accuracy estimation without access to labeled target data (Garg et al., 2022b).

We train models till the performance on validation source data (labeled) ceases to increase. Unlike OSDA methods, note that we do not use early stopping based on performance on held-out labeled target data. To evaluate classification performance, we report target accuracy on all classes, seen classes and the novel class. For target marginal, we separately report estimation error for previously seen classes and for the novel class. For the novel class, we report absolute difference between true and estimated marginal. For seen classes, we report average absolute estimation error. We open-source our code at `https://github.com/Neurips2022Anon`. By simply changing a single config file, new OSLS setups can be generated and experimented with.

[5] `https://github.com/mireshghallah/20Newsgroups-Pytorch`

Note that for our main experiments, for vision datasets (i.e., CIFAR10, CIFAR100, and Entity30) and for language dataset, we do not initialize with a (supervised) pre-trained model to avoid overlap of novel classes with the classes in the dataset used for pre-training. For example, labeled Imagenet-1k is typically used for pre-training. However, Imagenet classes overlaps with all three vision datasets employed and hence, we avoid pre-trained initialization. In App. D.6.7, we experiment with contrastive pre-training on Entity30 and CIFAR100. In contrast, for medical datasets, we leverage Imagenet pre-trained models as there is no overlap between classes in BreakHis and Dermnet with Imagenet.

D.6.4 Detailed results from main paper

For completeness, we next include results for all datasets. In particular, for each dataset we tabulate (i) overall accuracy on target; (ii) accuracy on seen classes in target; (iii) accuracy on the novel class; (iv) sum of absolute error in estimating target marginal among previously seen classes, i.e., $\sum_{y \in \mathcal{Y}_s} |\widehat{w} p_t(y) - p_t(y)|$; and (v) absolute error for novel fraction estimation, i.e., $|\widehat{w} p_t(y = k+1| - p_t(y = k+1)$. Table **??** presents results on all the datasets. Fig. **??** and Fig. **??** presents epoch-wise results.

D.6.5 Investigation into OSDA approaches

We observe that with default hyperparameters, popular OSDA methods significantly under perform as compared to PULSE. We hypothesize that the primary reasons underlying the poor performance of OSDA methods are (i) the heuristics employed to detect novel classes; and (ii) loss functions incorporated to improve alignment between examples from common classes in source and target. To detect novel classes, a standard heuristic employed popular OSDA methods involves thresholding uncertainty estimates (e.g., prediction entropy, softmax confidence (Fu et al., 2020; Saito et al., 2020; You et al., 2019)) at a predefined threshold κ. However, a fixed κ, may not for different datasets and different fractions of the novel class. Here, we ablate by (i) removing loss function terms incorporated with an aim to improve source target alignment; and (ii) vary threshold κ and show improvements in performance of these methods.

For our investigations, we experiment with CIFAR10, with UAN and DANCE methods. For DANCE, we remove the entropy separation loss employed to encourage align target examples with source examples. For UAN, we remove the adversarial domain discriminator training employed to align target examples with source examples. For both the methods, we observe that by removing the corresponding loss function terms we obtain a marginal improvement. For DANCE on CIFAR10, the performance goes up from 70.4 to 72.5 (with the same hyperparameters as the default run). FOR UAN, we observe similar minor improvements, where the performance goes up from 15.4 to 19.6.

Next, we vary the threshold used for detecting the novel examples. By optimally tuning the threshold for CIFAR10 with UAN, we obtain a substantial increase. In particular, the overall target accuracy increases from 19.6 to 33.1. With DANCE on CIFAR10, optimal threshold achieves 75.6 as compared to the default accuracy 70.4. In contrast, our two-stage method PULSE avoids the need to guess κ, by first estimating the fraction of novel class which then guides the classification of novel class versus previously seen classes.

D.6.6 Ablation with novel class fraction

In this section, we ablate on novel class proportion on CIFAR10, CIFAR100 and Newsgroups20. For each dataset we experiment with three settings, each obtained by varying the number of classes from the original data that constitutes the novel classes. We tabulate our results in Table **??**.

D.6.7 Contrastive pre-training on unlabeled data

Here, we experiment with contrastive pre-training to pre-train the backbone networks used for feature extraction. In particular, we initialize the backbone architectures with SimCLR pre-trained weights. We experiment with CIFAR100 and Entity30 datasets. Instead of pre-training on mixture of source and target unlabeled data, we leverage the publicly available pre-trained weights[6]. Table D.1 summarizes our results. We observe that pre-training improves over random initialization for all the methods with PULSE continuing to outperform other approaches.

Table D.1: Comparison with different OSLS approaches with pre-trained feature extractor. We use SimCLR pre-training to initialize the feature extractor for all the methods. All methods improve over random initialization (in Table 5.1). Note that PULSE continues to outperform other approaches.

Method	**CIFAR100**		**Entity30**	
	Acc (All)	MPE (Novel)	Acc (All)	MPE (Novel)
BODA (Saito et al., 2018b)	37.1	0.34	52.1	0.376
Domain Disc.	49.4	0.041	57.4	0.024
kPU	37.5	0.297	70.1	0.32
PULSE (Ours)	67.3	0.052	72.4	0.002

D.6.8 Ablation with different PU learning methods

In this section, we experiment with alternative PU learning approaches for PULSE and kPU. In particular, we experiment with the next best alternatives, i.e., nnPU instead of CVIR for classification and DEDPUL instead of BBE for target marginal estimation. We refer to these as kPU (alternative) and PULSE (alternative) in Table **??**. We present results

[6] For CIFAR100: https://drive.google.com/file/d/1huW-ChBVvKcx7t8HyDaWTQB5Li1Fht9x/view and for Entity30, we use Imagenet pre-trained weights from here: https://github.com/AndrewAtanov/simclr-pytorch.

on three datasets: CIFAR10, CIFAR100 and Newsgroups20 in the same setting as described in Sec. D.6.2. We make two key observations: (i) PULSE continues to dominate kPU with alternative choices; (ii) CVIR and BBE significantly outperform alternative choices.

D.6.9 Age Prediction Task

We consider an experiment on UTK Face dataset[7]. We create an 8-way class classification problem where we split the age in the following 8 groups: 0–10, 11–20, $\cdots$, 60–70 and > 70. We consider the first 7 age groups in source and introduce age group > 70 into the target data. OSLS continues to outperform the kPU baseline for novel prevalence estimation. Additionally, for target classification performance of OSLS is similar to kPU baseline (ref. Table D.2).

Table D.2: Results on age prediction dataset. We observe that the prevalence of the novel class as estimated with our PULSE framework is significantly closer to the true estimate. Additionally target classification performance of OSLS is similar to that of kPU both of which significantly improve over domain discriminator and source only baselines.

	UTK Face	
Method	Acc (All)	MPE (Novel)
Source Only	50.1	0.11
Domain Disc.	52.4	0.08
kPU	56.7	0.11
PULSE (Ours)	56.8	0.01

[7] https://susanqq.github.io/UTKFace/

Appendix E

Appendix: Complementary Benefits of Contrastive Learning and Self-Training Under Distribution Shift

E.1 Other Related Works

Unsupervised domain adaption. One line of research focuses on constructing benchmarks to develop heuristics for incorporating the unlabeled target data, relying on benchmark datasets ostensibly representative of "real-world shifts" to adjudicate progress (Peng et al., 2017; 2019; Sagawa et al., 2021; Santurkar et al., 2021; Venkateswara et al., 2017). As a result, various benchmark-driven heuristics have been proposed (Ganin et al., 2016; Long et al., 2015; 2017; Sohn et al., 2020; Sun and Saenko, 2016; Sun et al., 2017; Zhang et al., 2018c; 2019). Our work engages with the latter, focusing on two popular methods: self-training and contrastive pretraining.

Domain generalization. In domain generalization, the model is given access to data from multiple different domains and the goal is to generalize to a previously unseen domain at test time (Blanchard et al., 2011; Muandet et al., 2013). For a survey of different algorithms for domain generalization, we refer the reader to Gulrajani and Lopez-Paz (2020). A crucial distinction here is that unlike the domain generalization setting, in DA problems, we have access to unlabeled examples from the test domain.

Semi-supervised learning. To learn from a small amount of labeled supervision, semi-supervised learning methods leverage unlabeled data alongside to improve learning models. One of the seminal works in SSL is the pseudolabeling method (Scudder, 1965), where a classifier is trained on the labeled data and then used to classify the unlabeled data, which are then added to the training set. The work of Zhu and Ghahramani (2003) built on this by introducing graph-based methods, and the transductive SVMs (Joachims et al., 1999) presented an SVM-based approach. More recent works have focused on deep learning techniques, and similar to UDA, self-training and contrastive pretraining have emerged as two prominent choices. We delve into these methods in greater detail in the following paragraphs. For a discussion on other SSL methods, we refer interested readers to (Chapelle et al., 2006; Van Engelen and Hoos, 2020; Yang et al., 2022).

Self-training. Two popular forms of self-training are pseudolabeling (Lee et al., 2013) and conditional entropy minimization (Grandvalet and Bengio, 2006), which have been observed to be closely connected (Berthelot et al., 2019; Lee et al., 2013; Shu et al., 2018; Sohn et al., 2020). Motivated by its strong performance in SSL and UDA settings (Garg et al., 2023a; Shu et al., 2018; Sohn et al., 2020; Xie et al., 2020a), several theoretical works have made attempts to understand its behavior (Chen et al., 2020b; Kumar et al., 2020; Wei et al., 2020). (Cai et al., 2021a; Wei et al., 2020) aims to understand the behavior of the global minimizer of self-training objective by studying input consistency regularization, which enforces stability of the prediction for different augmentations of the unlabeled data. Our analysis of self-training is motivated by the work of Chen et al. (2020b) which explores the iterative behavior of self-training to unlearn spurious features. The setting of spurious features is of particular interest, since prior works have specifically analyzed the failures of out-of-distribution generalization in the presence of spurious features (Nagarajan et al., 2020; Sagawa et al., 2020).

Contrastive learning. An alternate line of work that uses unlabeled data for learning representations in the pretraining stage is contrastive learning (Caron et al., 2020; Chen et al., 2020a; Grill et al., 2020; Oord et al., 2018; Wu et al., 2018). Given an augmentation distribution, the main goal of contrastive objectives is to map augmentations drawn from the same input (positive pairs) to similar features, and force apart features corresponding to augmentations of different inputs (negative pairs) (Caron et al., 2020; 2021; He et al., 2020). Prior works (Cabannes et al., 2023; HaoChen and Ma, 2022; Johnson et al., 2022) have also shown a close relationship between contrastive (Chen et al., 2020a; HaoChen et al., 2021) and non-contrastive objectives (Bardes et al., 2021; Zbontar et al., 2021). Consequently, in our analysis pertaining to the toy setup we focus on the mathematically non-contrastive objective Barlow Twins (Zbontar et al., 2021). Using this pretrained backbone (either as an initialization or as a fixed feature extractor) a downstream predictor is learned using labeled examples. Several works (Arora et al., 2019b; HaoChen and Ma, 2022; HaoChen et al., 2021; Johnson et al., 2022; Saunshi et al., 2022) have analyzed the in-distribution generalization of the downstream predictor via label consistency arguments on the graph of positive pairs (augmentation graph). In contrast, we study the impact of contrastive learning under distribution shifts in the UDA setup. Other works (HaoChen et al., 2022; Shen et al., 2022) that examine contrastive learning for UDA also conform to the augmentation graph view point, making additional assumptions that guarantee linear transferability. In our simplified setup involving spurious correlations, these abstract assumptions break easily when the augmentations are of a generic nature, akin to practice. Finally, some empirical works (Ma et al., 2021; Mishra et al., 2021) have found self-supervised objectives like contrastive pretraining to reduce dependence on spurious correlations. Corroborating their findings, we extensively evaluate the complementary benefits of contrastive learning and self-training on real-world datasets. Finding differing results in SSL and UDA settings, we further examine their behavior theoretically in our toy setup.

E.2 More Details on Problem Setup

In this section, we elaborate on our setup and methods studied in our work.

Unsupervised Domain Adaptation (UDA). We assume that we are given labeled data from the *source* distribution and unlabeled data from a shifted, *target* distribution, with the goal of performing well on target data. We assume that the source and target distributions have the same label marginals $\mathrm{P}_s(y) = \mathrm{P}_t(y)$ (*i.e.*, no label proportion shift) and the same Bayes optimal predictor, *i.e.*, $\arg\max_y p_s(y \mid x) = \arg\max_y p_t(y \mid x)$. Here, even with infinite labeled source data, the challenge lies in generalizing out-of-distribution. In experiments, we assume access to finite data but in theory, we assume population access to labeled source and unlabeled target.

Semi-Supervised Learning (SSL). Here, there is no distribution shift, *i.e.*, $\mathrm{P}_s = \mathrm{P}_t = \mathrm{P}_{\mathsf{U}}$. We are given a small number of labeled examples and a comparatively large amount of unlabeled examples, both drawn from the same distribution. Without loss of generality, we denote this distribution with P_t. The goal in SSL is to generalize in-distribution. The challenge is primarily due to limited access to labeled data. Here, in experiments, we assume limited access to labeled data but a comparatively larger amount of unlabeled in-distribution data. In theory, we assume population access to unlabeled data but limited labeled examples.

Methods. As discussed in the main paper, we compare four methods for learning with labeled and unlabeled data. Table E.6 summarizes the main methods and key differences between those methods in UDA and SSL setup. For exact implementation in our experiments, we refer reader to App. E.3.3.

E.3 Additional Experiments and Details

E.3.1 Additional setup and notation

Recall, our goal is to learn a predictor that maps inputs $x \in \mathcal{X} \subseteq \mathbb{R}^d$ to outputs $y \in \mathcal{Y}$. We parameterize predictors $f = h \circ \Phi : \mathbb{R}^d \mapsto \mathcal{Y}$, where $\Phi : \mathbb{R}^d \mapsto \mathbb{R}^k$ is a feature map and $h \in \mathbb{R}^k$ is a classifier that maps the representation to the final scores or logits. With $A : \mathcal{X} \to \mathcal{A}$, we denote the augmentation function that takes in an input x and outputs an augmented view of the input $A(x)$. Unless specified otherwise, we perform full-finetuning in all of our experiments on real-world data. That is, we backpropagate gradients in both the linear head h and the backbone ϕ. For UDA, we denote source labeled points as $\{(x_i, y_i)\}_{i=1}^n$ and target unlabeled points as $\{(x_i')\}_{i=1}^m$. For SSL, we use the same notation for labeled and unlabeled in-distribution data.

E.3.2 Dataset details

For both UDA and SSL, we conduct experiments across eight benchmark datasets. Each of these datasets consists of domains, enabling us to construct source-target pairs for UDA. The adopted source and target domains are standard to previous studies (Garg et al., 2023a; Sagawa et al., 2021; Shen et al., 2022). Because the SSL setting lacks distribution shift, we do not need to worry about domain designations and default to using source alone. To simulate limited supervision in SSL, we sub-sample the original labeled training set to 10%. Below provide exact details about the datasets used in our benchmark study.

- **CIFAR10** We use the original CIFAR10 dataset (Krizhevsky and Hinton, 2009) as the source dataset. For target domains, we consider CINIC10 (Darlow et al., 2018) which is a subset of Imagenet restricted to CIFAR10 classes and downsampled to 32×32.

- **FMoW** In order to consider distribution shifts faced in the wild, we consider FMoW-WILDs (Christie et al., 2018; Koh et al., 2021) from WILDS benchmark, which contains satellite images taken in different geographical regions and at different times. We use the original train as source and OOD val and OOD test splits as target domains as they are collected over different time-period. Overall, we obtain 3 different domains (1 source and 2 targets).

- **BREEDs** We also consider BREEDs benchmark (Santurkar et al., 2021) in our setup to assess robustness to subpopulation shifts. BREEDs leverage class hierarchy in ImageNet (Russakovsky et al., 2015) to re-purpose original classes to be the subpopulations and defines a classification task on superclasses. We consider distribution shift due to subpopulation shift which is induced by directly making the subpopulations present in the training and test distributions disjoint. BREEDs benchmark contains 4 datasets **Entity-13**, **Entity-30**, **Living-17**, and **Non-living-26**, each focusing on different subtrees and levels in the hierarchy. Overall, for each of the 4 BREEDs datasets (i.e., Entity-13, Entity-30, Living-17, and Non-living-26), we obtain one different domain which we consider as target. We refer to source and target as follows: BREEDs sub-population 1, BREEDs sub-population 2.

- **OfficeHome** We use four domains (art, clipart, product and real) from OfficeHome dataset (Venkateswara et al., 2017). We use the product domain as source and the other domains as target.

- **Visda** We use three domains (train, val and test) from the Visda dataset (Peng et al., 2017; 2018). While 'train' domain contains synthetic renditions of the objects, 'val' and 'test' domains contain real world images. To avoid confusing, the domain names with their roles as splits, we rename them as 'synthetic', 'Real-1' and 'Real-2'. We use the synthetic (original train set) as the source domain and use the other domains as target.

We summarize the information about source and target domains in Table E.1.

Train-test splits We partition each source and target dataset into 80% and 20% i.i.d. splits. We use 80% splits for training and 20% splits for evaluation (or validation). We

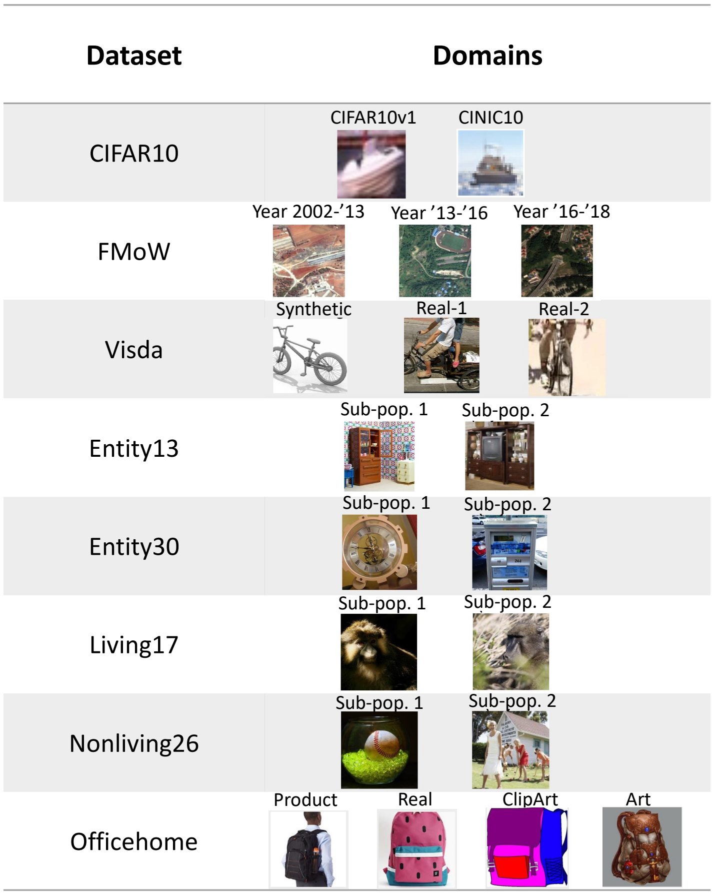


Figure E.1: Examples from all the domains in each dataset.

throw away labels for the 80% target split and only use labels in the 20% target split for final evaluation. The rationale behind splitting the target data is to use a completely unseen batch of data for evaluation. This avoids evaluating on examples where a model potentially could have overfit. over-fitting to unlabeled examples for evaluation. In practice, if the aim is to make predictions on all the target data (i.e., transduction), we can simply use the (full) target set for training and evaluation.

Simulating SSL settings and limited supervision. For SSL settings, we choose the in-distribution domain as the source domain. To simulate limited supervision in SSL, we sub-sample the original labeled training set to 10% and use all the original dataset as unlabeled data. For evaluation, we further split the original holdout set into two partitions (one for validation and the other to report final accuracy numbers).

E.3.3 Method details

For implementation, we build on top of WILDs (Sagawa et al., 2021) and RLSbench (Garg et al., 2023a) open source libraries.

Dataset	Source	Target
CIFAR10	CIFAR10v1	CINIC10
FMoW	FMoW (2002–'13)	FMoW (2013–'16), FMoW (2016–'18)
Entity13	Entity13 (sub-population 1)	Entity13 (sub-population 2)
Entity30	Entity30 (sub-population 1)	Entity30 (sub-population 2),
Living17	Living17 (sub-population 1)	Living17 (sub-population 2),
Nonliving26	Nonliving26 (sub-population 1)	Nonliving26 (sub-population 2),
Officehome	Product	Product, Art, ClipArt, Real
Visda	Synthetic (originally referred to as train)	Synthetic, Real-1 (originally referred to as val), Real-2 (originally referred to as test)

Table E.1: Details of source and target sets in each dataset considered in our testbed.

ERM (Source only) training. We consider Empirical Risk Minimization (ERM) on the labeled source data as a baseline. Since this simply ignores the unlabeled target data, we call this as source only training. As mentioned in the main paper, we perform source only training with data augmentations. Formally, we minimize the following ERM loss:

$$L_{\text{source only}}(f) = \frac{1}{n}\sum_{i=1}^{n} \ell(f(A(x_i), y_i))\,, \tag{E.1}$$

where A is the stochastic data augmentation operation and ℓ is a loss function. For SSL, the ERM baseline only uses the small of labeled data available.

Contrastive Learning (CL). We perform contrastive pretraining on the unlabeled dataset to obtain the backbone ϕ_{cl}. And then we perform full fine-tuning with source labeled data by initializing the backbone with ϕ_{cl}. We use SwAV (Caron et al., 2020) for contrastive pretraining. The main idea behind SwAV is to train a model to identify different views of the same image as similar, while also ensuring that it finds different images to be distinct. This is accomplished through a *swapped* prediction mechanism, where the goal is to compute a code from an augmented version of the image and predict this code from other augmented versions of the same image. In particular, given two image features $\phi(x'_{a1})$ and $\phi(x'_{a2})$ from two different augmentations of the same image x', i.e., $x'_{a1}, x'_{a2} \sim A(x')$, SwAV computes their codes z_{a1} and z_{a2} by matching the features to a set of K prototypes $\{c_1, \cdots, c_K\}$. Then SwAV minimizes the following loss such that $\phi(x'_{a1})$ can compute codes z_{a2} and $\phi(x'_{a2})$ can compute codes z_{a1}:

$$L_{\text{SwAV}}(\phi) = \sum_{i=1}^{m} \sum_{x'_{i,a1}, x'_{i,a2} \sim A(x'_i)} \ell'(\phi(x'_{i,a1}), z_{i,a2}) + \ell'(\phi(x'_{i,a2}), z_{i,a1})\,, \tag{E.2}$$

where ℓ' computes KL-divergence between codes computed with features (e.g. $\phi(x_{a1})$) and the code computed by another view (e.g. z_{a2}). For more details about the algorithm, we refer the reader to Caron et al. (2020). In all UDA settings, unless otherwise specified, we pool all the (unlabeled) data from the source and target to perform SwAV. For SSL, we leverage in-distribution unlabeled data.

We employ SimCLR (Chen et al., 2020a) for the CIFAR10 dataset, aligning with previous studies that have utilized contrastive pretraining on the same dataset (Kumar et al., 2022b; Shen et al., 2022). The reason for this choice is that SwAV relies on augmentations that involve cropping images to a smaller resolution, making it more suitable for datasets with larger resolutions beyond 32 × 32.

Self-Training (ST). For self-training, we apply FixMatch (Sohn et al., 2020), where the loss on labeled data and on pseudolabeled unlabeled data are minimized simultaneously. Sohn et al. (2020) proposed FixMatch as a variant of the simpler Pseudo-label method (Lee et al., 2013). This algorithm dynamically generates psuedolabels and overfits on them in each batch. FixMatch employs consistency regularization on the unlabeled data. In particular, while pseudolabels are generated on a weakly augmented view of the unlabeled examples, the loss is computed with respect to predictions on a strongly augmented view. The intuition behind such an update is to encourage a model to make predictions on weakly augmented data consistent with the strongly augmented example. Moreover, FixMatch only overfits to the assigned labeled with weak augmentation if the confidence of the prediction with strong augmentation is greater than some threshold τ. Refer to A_{weak} as the weak-augmentation and A_{strong} as the strong-augmentation function. Then, FixMatch uses the following loss function:

$$\begin{aligned} L_{\text{FixMatch}}(f) = &\frac{1}{n}\sum_{i=1}^{n} \ell(f(A_{\text{strong}}(x_i), y_i)) \\ &+ \frac{\lambda}{m}\sum_{i=1}^{m} \ell(f(A_{\text{strong}}(x_i'), \widetilde{y}_i)) \cdot \mathbb{I}\left[\max_y f_y(A_{\text{strong}}(x_i')) \geqslant \tau\right], \end{aligned}$$

where $\widetilde{y}_i = \arg\max_y f_y(T_{\text{weak}}(x_i))$. For UDA, our unlabeled data is the union of source and target unlabeled data. For SSL, we only leverage in-distribution unlabeled data.

We adapted our implementation from Sagawa et al. (2021) which matches the implementation of Sohn et al. (2020) except for one detail. While Sohn et al. (2020) augments labeled examples with weak augmentation, Sagawa et al. (2021) proposed to strongly augment the labeled source examples.

Self-Training Over Contrastive learning (STOC). Finally, rather than performing FixMatch from a randomly initialized backbone, we initialize FixMatch with a contrastive pretrained backbone.

E.3.4 Additional UDA experimemts

Table E.2: *Results in the UDA setup.* We report accuracy on target (OOD) data from which we only observe unlabeled examples during training. For benchmarks with multiple target distributions (*e.g.*, OH, Visda), we report average accuracy on those targets.

Method	Living17	Nonliv26	Entity13	Entity30	FMoW (2 tgts)	Visda (2 tgts)	OH (3 tgts)	CIFAR→ CINIC
ERM	60.2±0.1	45.4±0.2	68.6±0.1	55.7±0.0	56.5±0.1	20.8±0.2	9.5±0.2	74.3±0.1
ST	71.1±0.2	56.8±0.1	78.0±0.3	66.7±0.1	56.9±0.4	39.1±0.1	11.1±0.1	78.3±0.3
CL	74.1±0.2	57.4±0.3	76.9±0.2	66.6±0.3	61.5±0.5	63.2±0.2	22.8±0.1	77.5±0.1
STOC (ours)	**82.6±0.1**	**62.1±0.2**	**81.9±0.2**	**72.0±0.2**	**65.3±0.1**	**70.1±0.2**	**27.1±0.3**	**79.9±0.3**

Table E.3: *Results in the UDA setup with source only contrastive pretraining.* We report accuracy on target (OOD) data from which we only observe unlabeled examples during training. For benchmarks with multiple target distributions (*e.g.*, OH, Visda), we report average accuracy on those targets.

Method	Living17	Nonliv26	Entity13	Entity30	FMoW (2 tgts)	Visda (2 tgts)	OH (3 tgts)	CIFAR→ CINIC
CL (source only)	67.3±0.1	49.1±0.2	71.5±0.1	58.5±0.3	53.9±0.1	33.3±0.2	21.7±0.1	77.7±0.1
STOC (source only)	75.0±0.2	58.4±0.1	79.8±0.3	67.5±0.1	56.3±0.4	42.7±0.1	25.7±0.1	77.8±0.1
CL	74.1±0.2	57.4±0.3	76.9±0.2	66.6±0.3	61.5±0.5	63.2±0.2	22.8±0.1	77.5±0.1
STOC	**82.6±0.1**	**62.1±0.2**	**81.9±0.2**	**72.0±0.2**	**65.3±0.1**	**70.1±0.2**	**27.1±0.3**	**79.9±0.3**

E.3.5 Additional SSL experimemts

Table E.4: *Results in the SSL setup.* We report accuracy on hold-out ID data. Recall that SSL uses labeled and unlabeled data from the same distribution during training.

Method	Living17	Nonliv26	Entity13	Entity30	FMoW	Visda	OH	CIFAR
ERM	76.8±0.1	64.9±0.2	80.1±0.0	70.4±0.3	33.6±0.4	99.2±0.0	32.0±0.2	85.5±0.1
ST	85.4±0.1	75.7±0.2	85.4±0.2	77.3±0.1	33.6±0.3	99.2±0.1	32.0±0.1	93.1±0.1
CL	91.1±0.5	84.6±0.6	90.7±0.4	85.5±0.3	43.1±0.2	97.6±0.3	49.7±0.2	91.7±0.2
STOC (ours)	92.0±0.1	85.8±0.2	91.3±0.3	86.1±0.2	44.4±0.1	97.7±0.2	49.9±0.2	93.06±0.3

E.3.6 Other experimental details

Augmentations. For weak augmentation, we leverage random horizontal flips and random crops of pre-defined size. For SwAV, we also perform multicrop augmentation as proposed in Caron et al. (2020). For strong augmentation, we apply the following transformations sequentially: random horizontal flips, random crops of pre-defined size, augmentation with Cutout (DeVries and Taylor, 2017), and RandAugment (Cubuk et al., 2020). For the exact implementation of RandAugment, we directly use the implementation of Sohn et al. (2020). Unless specified otherwise, for all methods, we default to using strong augmentation techniques.

Architectures. In our work, we experiment with Resnet18, Resnet50 (He et al., 2016) trained from scratch (*i.e.* random initialization). We do not consider off-the-shelf pretrained models (*e.g.*, on Imagenet (Russakovsky et al., 2015)) to avoid confounding our conclusions about contrastive pretraining. However, we note that our results on most datasets tend to be comparable to and sometimes exceed those obtained with ImageNet pretrained models. For BREEDs datasets, we employ Resnet18 architecture. For other datasets, we train a Resnet50 architecture.

Except for Resnets on CIFAR dataset, we used the standard pytorch implementation (Gardner et al., 2018). For Resnet on Cifar, we refer to the implementation here: `https://github.com/kuangliu/pytorch-cifar`. For all the architectures, whenever applicable, we add antialiasing (Zhang, 2019). We use the official library released with the paper.

Hyperparameters. For all the methods, we fix the algorithm-specific hyperparameters to the original recommendations. For UDA, given that the setup precludes access to labeled data from the target distribution, we use source hold-out performance to pick the best hyperparameters. During pretraining, early stopping is done according to lower values of pretraining loss.

We tune the learning rate and ℓ_2 regularization parameter by fixing the batch size for each dataset that corresponds to the maximum we can fit to 15GB GPU memory. We default to using cosine learning rate schedule (Loshchilov and Hutter, 2016). We set the number of epochs for training as per the suggestions of the authors of respective benchmarks. For SSL, we run both ERM and FixMatch for approximately 2000 epochs. Note that we define the number of epochs as a full pass over the labeled training source data. We summarize the learning rate, batch size, number of epochs, and ℓ_2 regularization parameter used in our study in Table F.6.

Compute infrastructure. Our experiments were performed across a combination of Nvidia T4, A6000, and V100 GPUs.

E.4 Additional Results in Toy Setup

In this section we will first give more details on our simplified setup that captures both contrastive pretraining and self-training in the same framework. Then, we provide some

Dataset	Batch size	ℓ_2 regularization set	Learning rate set
CIFAR10	200	$\{0.001, 0.0001, 10^{-5}, 0.0\}$	$\{0.2, 0.1, 0.05, 0.01, 0.003, 0.001\}$
FMoW	64	$\{0.001, 0.0001, 10^{-5}, 0.0\}$	$\{0.01, 0.003, 0.001, 0.0003, 0.0001\}$
Entity13	256	$\{0.001, 0.0001, 10^{-5}, 0.0\}$	$\{0.4, 0.2, 0.1, 0.05, 0.02, 0.01, 0.005\}$
Entity30	256	$\{0.001, 0.0001, 10^{-5}, 0.0\}$	$\{0.4, 0.2, 0.1, 0.05, 0.02, 0.01, 0.005\}$
Entity30	256	$\{0.001, 0.0001, 10^{-5}, 0.0\}$	$\{0.4, 0.2, 0.1, 0.05, 0.02, 0.01, 0.005\}$
Nonliving26	256	$\{0.001, 0.0001, 10^{-5}, 0.0\}$	$\{0.4, 0.2, 0.1, 0.05, 0.02, 0.01, 0.005\}$
Officehome	96	$\{0.001, 0.0001, 10^{-5}, 0.0\}$	$\{0.01, 0.003, 0.001, 0.0003, 0.0001\}$
Visda	96	$\{0.001, 0.0001, 10^{-5}, 0.0\}$	$\{0.03, 0.01, 0.003, 0.001, 0.0003\}$

Table E.5: Details of the batch size, learning rate set and ℓ_2 regularization set considered in our testbed.

additional empirical results that are not captured theoretically but mimic behaviors observed in real world settings, highlighting the richness of our setup.

E.4.1 Detailed description of our simplified setup

In this subsection, we will first re-iterate the problem setup in Sec. 12.4 and provide some comparisons between our setup and those in closely related works. We will then describe the four methods: ERM, ST, CL, and STOC, providing details on the exact estimates returned by these algorithms in the SSL and UDA settings.

Data distribution. We consider binary classification and model the inputs as consisting of two kinds of features: $x = [x_{\text{in}}, x_{\text{sp}}]$ where $x_{\text{in}} \in \mathbb{R}^{d_{\text{in}}}$ is the invariant feature that is predictive of the label across both source P_s and target P_t and $x_{\text{sp}} \in \mathbb{R}^{d_{\text{sp}}}$ is the spurious feature that is correlated with the label y only on the source domain P_s but uncorrelated with label y in P_t. Here, $x_{\text{in}} \in \mathbb{R}^{d_{\text{in}}}$ determines the label using the ground truth classifier $w^\star \sim \text{Unif}(\mathbb{S}^{d_{\text{in}}-1})$, and $x_{\text{sp}} \in \mathbb{R}^{d_{\text{sp}}}$ is strongly correlated with the label on source but random noise on target. Formally, we sample $\mathrm{y} \sim \text{Unif}\{-1, 1\}$ and generate inputs x conditioned on y as follows

$$\begin{aligned} \mathrm{P}_s :\ & x_{\text{in}} \sim \mathcal{N}(\gamma \cdot \mathrm{y}w^\star, \Sigma_{\text{in}}) \quad x_{\text{sp}} = \mathrm{y}\mathbf{1}_{d_{\text{sp}}} \\ \mathrm{P}_t :\ & x_{\text{in}} \sim \mathcal{N}(\gamma \cdot \mathrm{y}w^\star, \Sigma_{\text{in}}) \quad x_{\text{sp}} \sim \mathcal{N}(\mathbf{0}, \Sigma_{\text{sp}}), \end{aligned} \tag{E.3}$$

where γ is the margin afforded by the invariant feature. We set covariance of the invariant features $\Sigma_{\text{in}} = \sigma_{\text{in}}^2 \cdot (\mathbf{I}_{d_{\text{in}}} - w^\star w^{\star\top})$ to capture structure in the invariant feature that the variance is less along the latent predictive direction $w^\star$. Note that the spurious feature is completely predictive of the label in the source data, and is distributed as spherical Gaussian in the target data with $\Sigma_{\text{sp}} = \sigma_{\text{sp}}^2 \mathbf{I}_{d_{\text{sp}}}$.

Why is our simplified setup interesting? In our setup, x_{in} is the hard to learn feature that generalizes from source to target. The hardness of learning this feature is determined by the value of the margin γ and how it compares with size of the spurious feature ($\sqrt{d_{\text{sp}}}$). Since, $\gamma/\sqrt{d_{\text{sp}}}$ is small in our setup, x_{in} is much harder to learn on source data (even with population access) compared to the spurious feature x_{sp} which generalizes poorly from source to target. These two types of features have been captured in similar analysis on spurious correlations (Nagarajan et al., 2020; Sagawa et al., 2020) since it imitates pitfalls emanating from the presence of spurious features in real world datasets (*e.g.*, the easy to learn background feature in image classification problems). While this setup is simple, it is also expressive enough to elucidate both self-training and contrastive learning behaviors we observe in real world settings. Specifically, it captures the separation results we observe in Sec. 2.6.

Differences of our setup with prior works. While our distribution shift settings bears the above similarities it also has important differences with works analyzing self-training and contrastive pretraining individually. Chen et al. (2020b) analyze the iterative nature of self-training algorithm, where the premise is that we are given a classifier that not only has good performance on source data but in addition does not rely too much on the spurious feature. Under the strong condition of small norms along the spurious feature, they show that self-training can provably unlearn this small dependence when the target data along the spurious feature is random noise. This assumption is clearly violated in setups where the spurious correlation is strong (as in our toy setup), *i.e.*, the dependence on the spurious feature is rather large (much larger than that on the invariant feature) for any classifier that is trained directly on source data. Consequently, we show the need for "good" pretrained representations from contrastive pretraining over which if we train a linear predictor (using source labeled data), it will provably have a reduced "effective" dependence on the spurious feature.

Using an augmentation distribution similar to ours, Saunshi et al. (2022) carried out contrastive pretraining analysis with the backbone belonging to a capacity constrained function class (similar analysis also in (HaoChen et al., 2022)). Our setup differs from this in two key ways: (i) we specifically consider a distribution shift from source to target. Unlike their setting, it is not sufficient to make augmentations consistent with ground truth labels, since the predictor that uses just the spurious feature also assigns labels consistent with both ground truth predictions and augmentations on the source data; and (ii) our augmentation distribution assumes no knowledge of the invariant feature, which is why we augment all dimensions uniformly, as opposed to selectively augmenting a set of dimensions. In other words, we assume no knowledge of the structure of the optimal target predictor. For *e.g.*, if we had knowledge of the spurious dimensions we could have just selectively augmented those. Assuming knowledge of these perfect augmentations is not ideal for two reasons: (a) it makes the problem so easy that just training an ERM model on source data with these augmentations would already yield a good target predictor (which rarely happens in practice); and (b) in real-world datasets perfect augmentations for the downstream task are not known. Hence, we stick to generic augmentations in our setup.

E.4.2 Discussion on self-training and contrastive learning objectives

Method	**UDA Setup**	**SSL Setup**
ERM:	$h_{\text{erm}} = \arg\min_h \mathbb{E}_{\mathrm{P}_s} \ell(h(x), y)$	$h_{\text{erm}} = \arg\min_h \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i), y_i)$ $\{(x_i, y_i)\}_{i=1}^{n} \sim \mathrm{P}_t^n$
ST:	Starting from h_{erm} optimize over h (to get h_{st}): $\mathbb{E}_{\mathrm{P}_t(x)} \ell(h(x), \text{sgn}(h(x)))$	Starting from h_{erm} optimize over h (to get h_{st}): $\mathbb{E}_{\mathrm{P}_t(x)} \ell(h(x), \text{sgn}(h(x)))$
CL:	$\Phi_{\text{cl}} = \arg\min_\phi \mathcal{L}_{\text{cl}}(\Phi)$ Use $(\mathrm{P}_s(x) + \mathrm{P}_t(x))/2$ for $\mathcal{L}_{\text{cl}}(\Phi)$ $h_{\text{cl}} = \arg\min_h \mathbb{E}_{\mathrm{P}_s} \ell(h \circ \Phi_{\text{cl}}(x), y)$	$\Phi_{\text{cl}} = \arg\min_\phi \mathcal{L}_{\text{cl}}(\Phi)$ Use $\mathrm{P}_t(x)$ for $\mathcal{L}_{\text{cl}}(\Phi)$ $h_{\text{cl}} = \arg\min_h \frac{1}{n} \sum_{i=1}^{n} \ell(h \circ \Phi_{\text{cl}}(x_i), y_i)$
STOC:	Starting from h_{cl} optimize over h (to get h_{stoc}): $\mathbb{E}_{\mathrm{P}_t(x)} \ell(h \circ \Phi_{\text{cl}}(x), \text{sgn}(h \circ \Phi_{\text{cl}}(x)))$	Starting from h_{cl} optimize over h (to get h_{stoc}): $\mathbb{E}_{\mathrm{P}_t(x)} \ell(h \circ \Phi_{\text{cl}}(x), \text{sgn}(h \circ \Phi_{\text{cl}}(x)))$

Table E.6: **Description of methods for SSL vs. UDA**: For each method we provide exact objectives used for experiments and analysis in the SSL and UDA setups (pertaining to Sec. 12.4).

In text we will describe our objectives and methods for the UDA setup. In Table E.6 we constrast the differences in the methods and objectives for SSL and UDA setups. Recall from Section 6.2 that we learn linear classifiers h over features extractors Φ. We consider linear feature extractor i.e. Φ is a matrix in $\mathbb{R}^{k \times d}$. For mathematical convenience, we assume access to infinite unlabeled data and hence replace the empirical quantities over unlabeled data with their population counterpart. In the UDA setting, we further assume access to infinite labeled data from the source. Note that due to distribution shift between source and target, "ERM" on infinite labeled data from the source does not necessarily achieve optimal performance on the target. For binary classification, we assume that the linear layer h maps features to a scalar in $\mathbb{R}$ such that the prediction is $\text{sgn}(h^\top \Phi x)$. We use the exponential loss $\ell(f(x), y) = \exp(-y f(x))$ as the classification loss.

Contrastive pretraining. We obtain $\Phi_{\text{cl}} := \arg\min_\Phi \mathcal{L}_{\text{cl}}(\Phi)$ by minimizing the Barlow Twins objective (Zbontar et al., 2021), which prior works have shown is also equivalent to spectral contrastive and non-contrastive objectives (Cabannes et al., 2023; Garrido et al., 2022). In Sec. 12.4, we consider a constrained form of Barlow Twins in (6.3) which enforces representations of different augmentations a_1, a_2 of the same input x to be close in representation space, while ensuring feature diversity by staying in the constraint set. We assume a strict constraint on regularization ($\rho = 0$) for the theoretical arguments in the rest of the main paper. In App. E.5.1 we prove that all our claims hold for small ρ as well. In (E.4), we redefine the pretraining objective with a regularization term (instead of a constraint set) where κ controls the strength of the regularization term, with higher values of κ corresponding to stronger constraints on feature diversity. We then learn a linear classifier h_{cl} over Φ_{cl} to minimize the exponential loss on labeled source data.

$$\mathcal{L}_{\text{cl}}(\Phi) \; := \; \mathbb{E}_{x\sim \mathrm{P}_{\mathsf{U}}}\mathbb{E}_{a_1,a_2\sim \mathrm{P}_{\mathsf{A}}(\cdot|x)} \left\|\Phi(a_1) - \Phi(a_2)\right\|_2^2 \;\; + \;\; \kappa \cdot \left\|\mathbb{E}_{a\sim \mathrm{P}_{\mathsf{A}}}\left[\Phi(a)\Phi(a)^\top\right] - \mathbf{I}_k\right\| F^2 \tag{E.4}$$

Augmentations. Data augmentations play a key role in contrastive pre-training (and also as we see later, state-of-the-art self-training variants like FixMatch). Given input $x \in \mathcal{X}$, let $\mathrm{P}_{\mathsf{A}}(a \mid x)$ denote the distribution over its augmentations, and P_{A} denote the marginal distribution over all possible augmentations. We use the following simple augmentations where we scale the magnitude of each co-ordinate by a uniformly independent amount, *i.e.*,

$$a \sim \mathrm{P}_{\mathsf{A}}(\cdot \mid x) \equiv c \odot x \quad \text{where,} \quad c \sim \text{Unif}[0,1]^d. \tag{E.5}$$

The performance of different methods heavily depends on the assumptions we make on augmentations. We try to mirror practical settings where the augmentations are fairly "generic", not encoding any information about which features are invariant or spurious, and hence perturb all features symmetrically.

Self-training. ST performs ERM in the first stage using labeled data from the source, and then subsequently updates the head h by iteratively generating pseudolabels on the unlabeled target:

$$\mathcal{L}_{\text{st}}(h;\Phi) \; := \mathbb{E}_{\mathrm{P}_t(x)}\ell(h^\top \Phi x, \text{sgn}(h^\top \Phi(x))) \qquad \text{Update: } \; h^{t+1} = \frac{h^t - \eta\nabla_h \mathcal{L}_{\text{st}}(h^t;\Phi)}{\|h^t - \eta\nabla_h \mathcal{L}_{\text{st}}(h^t;\Phi)\| \, 2} \tag{E.6}$$

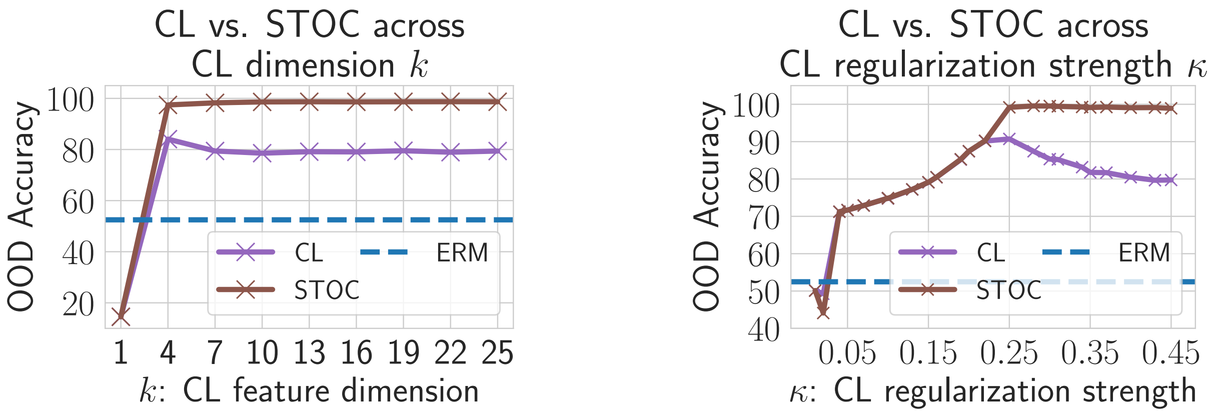
For convenience, we keep the feature backbone Φ fixed across the self-training iterations and only update the linear head on the pseudolabels.

STOC(Self-training after contrastive learning). Finally, we can combine the two unsupervised objectives where we do the self-training updates(6.2) with $h_0 = h_{\text{cl}}$ and $\Phi_0 = \Phi_{\text{cl}}$ starting with the contrastive learning model rather than just source-only ERM. Here, we only update h and fix Φ_{cl}.

E.4.3 Additional empirical results in our simplified setup

We conduct two ablations on the hyperparameters for contrastive pretraining. First, we vary the dimensionality k of the linear feature extractor $\Phi \in \mathbb{R}^{k\times d}$. Second, we vary the regularization strength κ that enforces feature diversity in the Barlow Twins objective (E.4). In Figure E.2 we plot these ablations in the UDA setup.

Varying feature dimension. We find that CL recovers the full set of predictive features (*i.e.* both spurious and invariant) only when k is large enough (Figure E.2*(left)*). Since the



(a)

(b)

Figure E.2: **Ablations on pretraining hyperparameters:** In the UDA setup we plot the performance of CL and STOC as we vary two pretraining hyper-parameters: *(left)* the output dimension (k) of the feature extractor Φ; and *(right)* the strength (κ) of the regularizer in the Barlow Twins objective in (E.4). While ablating on k we fix $\kappa = 0.5$, and while ablating on κ we fix $k = 10$. Other problem parameters are taken from Example 1.

dimensionality of the true feature is 5 in our Example 1, reducing k below the true feature dimension hurts CL. Once k crosses a certain threshold, CL features completely capture the projection of the invariant feature w_{in}. After this point, it amplifies the component along w_{in}. It retains the amplification over the spurious feature w_{sp} even as we increase k. This is confirmed by our finding that further increasing k does not hurt CL performance. This is also inline with our theoretical observations, where we find that for suitable $w^\star$, the subspace spanned by w_{in} and w_{sp} are contained in a low rank space (as low as rank 2) of the contrastive representations (Theorem 6.4.4). Once CL has amplified the dependence along w_{in} STOC improves over CL by unlearning any remaining dependence on the spurious w_{sp}. The above arguments for the CL trend also explain why the performance of STOC continues to remain $\approx 100\%$ as we vary k.

Varying regularization strength. In our main theoretical arguments we consider the constrained form of the Barlow Twins objective (6.3) with a strict constraint of $\rho = 0$ (we relax this theoretically as well, see E.5.1). For our experiments, we optimize the regularized version of this objective (E.4), where the constraint term now appears as a regularizer which enforces feature diversity, *i.e.* the features learned through contrastive pretraining span orthogonal parts of the input space (as governed under the metric defined by augmentation covariance matrix Σ_A). If κ is very low, then trivial solutions exist for the Barlow Twins objective. For *e.g.*, $\phi \approx \mathbf{0}$ (zero vector) achieves very low invariance loss. When $\kappa < 0.05$, we find that CL recovers these trivial solutions (Figure E.2*(right)*). Hence, both CL and STOC perform poorly. As we increase κ the performance of both CL and STOC improve, mainly because the features returned by Φ_{cl} now comprise of the predictive directions w_{in} and w_{sp}, as predictive by our theoretical arguments for $\rho = 0$ (which corresponds to large

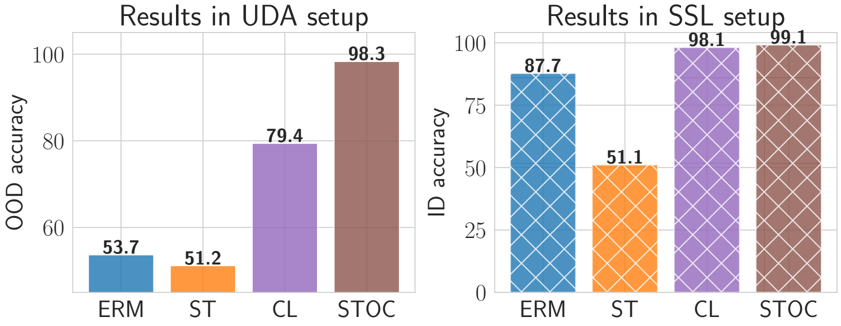


Figure E.3: **Results with linear backbone:** We plot the OOD accuracy for ERM, CL, ST and STOC in the UDA setup and ID accuracy in the SSL setup when the feature extractor Φ is a linear network. Note, that the feature extractor is still fixed during CL and STOC.

κ). On the other hand, when κ is too high optimization becomes hard since κ directly effects the Lipschitz constant of the loss function. Hence, the performance of CL drops by some value. Note that this does not effect the performance of STOC since CL continues to amplify w_{in} over w_{sp} even if it is returning suboptimal solutions with respect to the optimization loss of the pretraining objective.

E.4.4 Reconciling Practice: Experiments with deep networks in toy setup

In this section we delve into the details of Sec. 6.4.5, *i.e.*, we analyze performance of different methods when we make some design choices that imitate practice. First, we look at experiments involving a deep non-linear backbone Φ. Here, the non-linear Φ is learned during contrastive pretraining and fixed for CL and STOC. Then, we investigate trends when we continue to propagate gradients onto Φ during STOC (we call this full-finetuning). Unlike previous cases, this allows features to be updated.

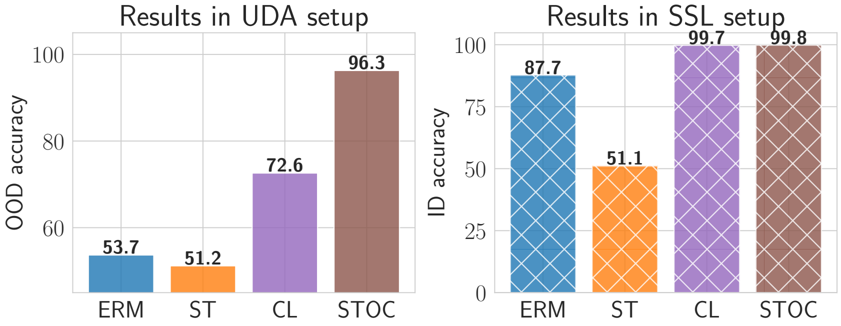
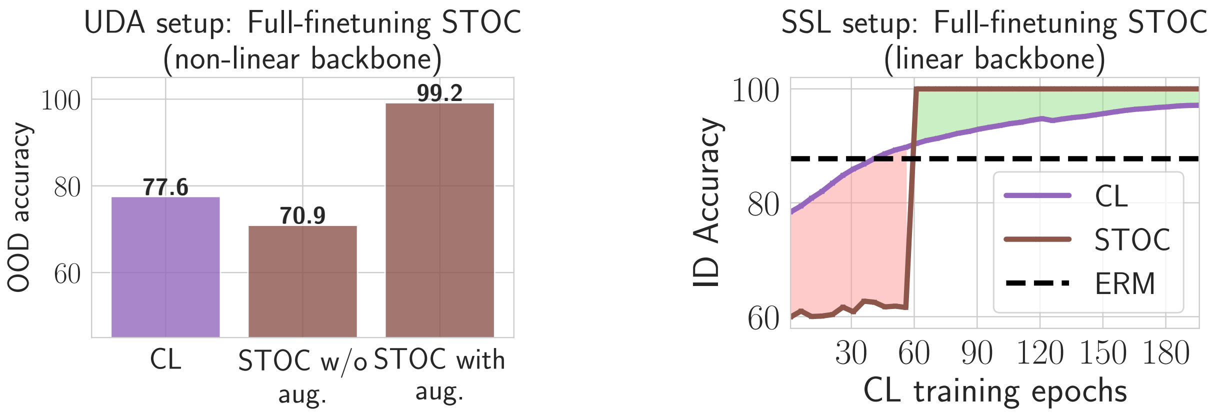


Figure E.4: **Results with non-linear backbone:** We plot the OOD accuracy for ERM, CL, ST and STOC in the UDA setup and ID accuracy in the SSL setup when the feature extractor Φ is a non-linear one-hidden layer network with ReLU activations. Note, that the feature extractor is still fixed during CL and STOC.



(a)

(b)

Figure E.5: **Finetuning the contrastive representations during STOC:** We propagate gradients to the feature backbone Φ when running STOC algorithm. Note that CL still fixes the contrastive representations when learning a fixed linear head over it. On the *(left)* we show results in UDA setup where we compare the performance of STOC with and without augmentations (along with other practical design choices like confidence thresholds and continuing to optimize source loss as done in FixMatch) when the feature backbone is non-linear. On the *(right)* we show results for STOC and CL in the SSL setup when the feature backbone is linear.

Results with non-linear feature extractor Φ. In Fig. E.4 we plot the performance of the four methods when we use a non-linear feature extractor during contrastive pretraining. This feature extractor is a one-hidden layer neural network (hidden dimension is 500) with ReLU activations. We find that the trends observed with linear backbones in Fig. E.3 are also replicated with the non-linear one. Specifically, we note that STOC improves over CL under distribution shifts, whereas CL is already close to optimal when there are no distribution shifts. We also see that CL and ST individually are subpar. In SSL, we see a huge drop in the performance of ST (over ERM) mainly because we only fit on pseudolabels during ST. This is different from practice where we continue to optimize loss on labeled data points while fitting the pseudolabels. Consequently, when we continue to optimize performance on source labeled data the performance of ST in SSL setup is improves from 51.1% → 72.6%.

Results with full fine-tuning. Up till this point, we have only considered the case (for both SSL and UDA) where we fix the contrastive learned features when running CL and STOC, *i.e.*, we only optimized the linear head h. Now, we shall consider the setting where gradients are propagated to Φ during STOC. Note that we still fix the representations for training the linear head during CL. Results for this setting are in Figure E.5. We show two interesting trends that imitate real world behaviors.

STOC benefits from augmentations during full-finetuning: In the UDA setup we find that ST while updating Φ_{cl} can hurt due to overfitting issues when training with the finite sample of labeled and unlabeled data (drop by $> 7\%$ over CL). This is due to overfitting on confident but incorrect pseudolabels on target data. This can exacerbate components along spurious feature w_{sp} from source. One reasoning behind this is that deep neural networks can perfectly memorize them on finite unlabeled target data (Zhang et al., 2017). Heuristics typically used in practice (*e.g.* in FixMatch (Sohn et al., 2020)) help avoid overfitting on incorrect pseudolabels: (i) confidence thresholding; to pick confident pseudolabel examples; (ii) pseudolabel a different augmented input than the one on which the self-training loss is optimized; and (iii) optimize source loss with labeled data simultaneously when fitting pseudolabels. Intuitively, thresholding introduces a curriculum where we only learn confident examples in the beginning whose pseudolabels are mainly determined by component along the invariant feature w_{in}. Augmentations prevent the neural network from memorizing incorrect pseudolabels and optimizing source loss prevents forgetting of features learned during CL. When we implement these during full-finetuning in STOC we see that STOC now improves over CL (by $> 20\%$).

Can we improve contrastive pretraining features during STOC? We find that self-training can also improve features learned during contrastive pretraining when we update the full backbone during STOC (see Figure E.5*(right)*). Specifically, in the SSL setup we find that STOC can now improve substantially over CL. Recall, that when we fixed Φ_{cl} this was not possible (see E.5.3 and Fig. 6.2(b)). This is mainly because STOC can now improve performance beyond just recovering the generalization gap for the linear head (which is typically small). This feature improvement is observed even when we fully finetune a linear feature extractor. Similar trends are also observed with the non-linear backbone. But, it becomes harder to identify a good stopping criterion for CL training. Thus, it remains unclear if STOC and CL have complementary benefits for feature learning in UDA or SSL settings. Investigating this is an interesting avenue for future work.

E.5 Formal Statements from Sec. 12.4

Recall from Section 6.2 that we learn linear classifiers h over features extractors Φ. We consider linear feature extractor i.e. Φ is a matrix in $\mathbb{R}^{d\times k}$ and the linear layer $h : \mathbb{R}^k \to \mathbb{R}$ with a prediction as $\text{sgn}(h^\top \Phi x)$. We use the exponential loss $\ell(f(x), y) = \exp(-yf(x))$.

E.5.1 Analysis of ERM and ST: Formal Statement of Theorem 6.4.2

For ERM and ST, we train both h and Φ. This is equivalent to $\Phi = I_{d\times d}$ being identity and training a linear head h. Recall that the ERM classifier is obtained by minimizing the population loss on labeled source data:

$$h_{\text{ERM}} = \arg\min_h \mathbb{E}_{(x,y)\sim \mathrm{P}_s}\left[\ell(x, y)\right] . \tag{E.7}$$

We split Theorem 6.4.2 into Theorem E.5.1 and Theorem E.5.2. Before we characterize the ERM solution, we recall some additional notation. Define $w_{\text{in}}=[w^\star, 0, ..., 0]^\top$, and

$w_{\text{sp}} = [0, ..., 0, {}^{\mathbf{1}_{d_{\text{sp}}}}/_{\sqrt{d_{\text{sp}}}}]^\top$. The following proposition characterizes h_{ERM} and 0-1 error of the classifier on target:

Theorem E.5.1 (ERM classifier and its error on target)**.** *ERM classifier obtained as in* (E.7) *is given by*

$$\frac{h_{ERM}}{\|h_{ERM}\|\,2} = \frac{\gamma \cdot w_{\text{in}} + \sqrt{d_{\text{sp}}} \cdot w_{\text{sp}}}{\sqrt{\gamma^2 + d_{\text{sp}}}} \,.$$

The target accuracy of h_{ERM} is given by $0.5 \cdot \text{erfc}\left(-{}^{\gamma^2}/_{(\sqrt{2d_{\text{sp}}} \cdot \sigma_{\text{sp}})}\right)$.

Proof. To prove this theorem, we first derive a closed-form expression for the ERM classifier and then use Lemma E.7.10 to derive its 0-1 error on target. For Gaussian data with the same covariance matrices for class conditional $\mathrm{P}_s(x|y=1)$ and $\mathrm{P}_s(x|y=0)$, Bayes decision rule is given by the Fisher's linear discriminant direction (Chapter 4; Bishop (2006)):

$$h(x) = \begin{cases} 1, & \text{if } h^\top x > 0 \\ 0, & \text{otherwise} \end{cases}$$

where $h = 2 \cdot \gamma(w_{\text{in}}) + 2 \cdot \sqrt{d_{\text{sp}}}(w_{\text{sp}})$. Plugging h in Lemma E.7.10 we get the desired result. □

ST performs ERM in the first stage using labeled data from the source, and then subsequently updates the head h by iteratively generating pseudolabels on the unlabeled target:

$$\mathcal{L}_{\text{st}}(h) \;\coloneqq \mathbb{E}_{\mathrm{P}_t(x)} \ell(h^\top x, \text{sgn}(h^\top x)) \,. \tag{E.8}$$

Starting with $h^0_{\text{ST}} = {}^{h_{\text{ERM}}}/_{\|h_{\text{ERM}}\|2}$ (the classifier obtained with ERM) we perform the following iterative procedure for self-training:

$$h^{t+1}_{\text{ST}} = \frac{h^t_{\text{ST}} - \eta \nabla_h \mathcal{L}_{\text{st}}(h^t_{\text{ST}})}{\|h^t_{\text{ST}} - \eta \nabla_h \mathcal{L}_{\text{st}}(h^t_{\text{ST}})\|\,2} \tag{E.9}$$

Next, we characterize ST solution:

Theorem E.5.2 (ST classifier and its error on target)**.** *Starting with ERM solution, ST will lead to:*

(i) (Necessary condition) $h^t_{ST} = w_{\text{sp}}$ as $t \to \infty$, such that the target accuracy is 50% for all $\sigma_{\text{sp}} \geqslant 1$ and $\gamma \leqslant \frac{1}{2\sqrt{\sigma_{\text{sp}}}}$.

(ii) (Sufficient condition) $h^t_{ST} = w_{\text{in}}$ as $t \to \infty$, such that the target accuracy is 100% when the problem parameters $\gamma, \sigma_{\text{sp}}$ satisfy: $\gamma \geqslant \sigma_{\text{sp}}$.

Proof. The proof can be divided into two parts: (i) deriving closed-form expressions for updates on h^t_{ST} in terms of h^{t-1}_{ST} and (ii) obtaining conditions under which the component along w_{in} monotonically increases or decreases with t after re-normalizing the norm of updated h. For notation convenience, we denote h_{ST} with h in the rest of the proof.

Part-1. First, the loss of self-training with classifier $h := [h_{\text{in}}, h_{\text{sp}}]$ where $h_{\text{in}} \in \mathbb{R}^{d_{\text{in}}}$ and $h_{\text{sp}} \in \mathbb{R}^{d_{\text{sp}}}$ is given by:

$$\begin{aligned}
\mathcal{L}_{\text{st}}(h) &= \mathbb{E}_{\mathrm{P}_t(x)}\left[\ell(h^\top x, \operatorname{sgn}(h^\top x))\right] && \text{(E.10)}\\
&= \mathbb{E}_{\mathrm{P}_t(x)}\left[\exp\left(-\operatorname{sign}(h^\top x)\cdot(h^\top x)\right)\right] && \text{(E.11)}\\
&= \mathbb{E}_{\mathrm{P}_t(x)}\left[\exp\left(-\left|h^\top x\right|\right)\right] && \text{(E.12)}\\
&= \mathbb{E}_{\mathrm{P}_t(x)}\left[\exp\left(-\left|h_{\text{in}}^\top x_{\text{in}} + h_{\text{sp}}^\top x_{\text{sp}}\right|\right)\right] && \text{(E.13)}\\
&= \mathbb{E}_{y\sim^U\{-1,1\}, z\sim\mathcal{N}(0,1)}\Big[\exp\Big(-\Big|\gamma\cdot y\cdot h_{\text{in}}^\top w^\star \\
&\qquad\qquad + \left[\sigma_{\text{in}}(\|h_{\text{in}}\|\, 2^2 - (h_{\text{in}}^T w^\star)^2) + \sigma_{\text{sp}}\cdot\|h_{\text{sp}}\|\, 2\right]\cdot z\Big|\Big)\Big]\,. && \text{(E.14)}\\
&= \mathbb{E}_{z\sim\mathcal{N}(0,1)}\left[\exp\left(-\left|\gamma\cdot h_{\text{in}}^\top w^\star + \left[\sigma_{\text{in}}(\|h_{\text{in}}\|\, 2^2 - (h_{\text{in}}^T w^\star)^2) + \sigma_{\text{sp}}\cdot\|h_{\text{sp}}\|\, 2\right]\cdot z\right|\right)\right]\,, && \text{(E.15)}
\end{aligned}$$

where (E.13) to (E.14) is implied by simply replacing the definition of target distribution and (E.14) to (E.15) is implied by the symmetry of the function with respect to y and $-y$ due to the symmetry of the absolute function and Gaussian distribution. For a classifier h^t, we denote $\mu_t = \gamma\cdot {h_{\text{in}}^t}^\top w^\star$ and $\sigma_t = \left[\sigma_{\text{in}}(\|h_{\text{in}}^t\|\, 2^2 - ({h_{\text{in}}^t}^T w^\star)^2) + \sigma_{\text{sp}}\cdot\left\|h_{\text{sp}}^t\right\|\, 2\right]$. With this notation, we can re-write the loss in (E.15) as $\mathcal{L}_{\text{st}}(h^t) = \mathbb{E}_{z\sim\mathcal{N}(0,\sigma_t^2)}\left[\exp\left(-\left|\mu_t + z\right|\right)\right]$.

Now we derive a closed-form expression of $\mathcal{L}_{\text{st}}(h^t)$ in Lemma E.7.11:

$$\mathcal{L}_{\text{st}}(h^t) = \frac{1}{2}\left(\exp\left(\frac{\sigma_t^2}{2} - \mu_t\right)\cdot\operatorname{erfc}\left(-\frac{\mu_t}{\sqrt{2}\sigma_t} + \frac{\sigma_t}{\sqrt{2}}\right) + \exp\left(\frac{\sigma_t^2}{2} + \mu_t\right)\cdot\operatorname{erfc}\left(\frac{\mu_t}{\sqrt{2}\sigma_t} + \frac{\sigma_t}{\sqrt{2}}\right)\right)\,. \tag{E.16}$$

Define the Mill's ratio as $\mathrm{r}(x) = \exp(x^2/2)\cdot\operatorname{erfc}\left(x/\sqrt{2}\right)\cdot\sqrt{\pi/2}$ as in Baricz (2008). We will frequently use standard properties of the Mill's ratio. We list them in Lemma E.7.2 for completeness. Define:

$$\begin{aligned}
\alpha_1(\mu_t,\sigma_t) &= -\exp\left(\frac{\sigma_t^2}{2} - \mu_t\right)\cdot\operatorname{erfc}\left(-\frac{\mu_t}{\sqrt{2}\sigma_t} + \frac{\sigma_t}{\sqrt{2}}\right) + \exp\left(\frac{\sigma_t^2}{2} + \mu_t\right)\cdot\operatorname{erfc}\left(\frac{\mu_t}{\sqrt{2}\sigma_t} + \frac{\sigma_t}{\sqrt{2}}\right)\,,\\
&= \sqrt{\frac{2}{\pi}}\exp\left(-\frac{\mu_t^2}{2\sigma_t^2}\right)\left[\mathrm{r}\left(\sigma_t + \frac{\mu_t}{\sigma_t}\right) - \mathrm{r}\left(\sigma_t - \frac{\mu_t}{\sigma_t}\right)\right] && \text{(E.17)}\\
\alpha_2(\mu_t,\sigma_t) &= \exp\left(\frac{\sigma_t^2}{2} - \mu_t\right)\cdot\operatorname{erfc}\left(-\frac{\mu_t}{\sqrt{2}\sigma_t} + \frac{\sigma_t}{\sqrt{2}}\right) + \exp\left(\frac{\sigma_t^2}{2} + \mu_t\right)\cdot\operatorname{erfc}\left(\frac{\mu_t}{\sqrt{2}\sigma_t} + \frac{\sigma_t}{\sqrt{2}}\right)\\
&\qquad - \frac{2\sqrt{2}}{\sigma_t\sqrt{\pi}}\exp\left(-\frac{\mu_t^2}{2\sigma_t^2}\right)\\
&= \sqrt{\frac{2}{\pi}}\exp\left(-\frac{\mu_t^2}{2\sigma_t^2}\right)\left[\mathrm{r}\left(\sigma_t + \frac{\mu_t}{\sigma_t}\right) + \mathrm{r}\left(\sigma_t - \frac{\mu_t}{\sigma_t}\right) - \frac{2}{\sigma_t}\right]\,. && \text{(E.18)}
\end{aligned}$$

Let $\widetilde{h}^{t+1}$ denote the un-normalized gradient descent update at iterate $t+1$. We have:

$$\widetilde{h}^{t+1} = h^t - \eta\cdot\frac{\partial\mathcal{L}_{\text{st}}(h^t)}{\partial h}\,. \tag{E.19}$$

Now we will individually argue about the update of $\widetilde{h}^{t+1}$ along the first d_{in} dimensions and the last d_{sp} dimensions. First, we have:

$$\begin{aligned}
\widetilde{h}_{\text{in}}^{t+1} &= h_{\text{in}}^{t} - \eta \cdot \frac{\partial \mathcal{L}_{\text{st}}(h^t)}{\partial h_{\text{in}}} \\
&= h_{\text{in}}^{t} - \frac{\eta}{2}\left(-\exp\left(\frac{\sigma_t^2}{2} - \mu_t\right) \cdot \operatorname{erfc}\left(-\frac{\mu_t}{\sqrt{2}\sigma_t} + \frac{\sigma_t}{\sqrt{2}}\right) \right. \\
&\qquad\qquad \left. +\exp\left(\frac{\sigma_t^2}{2} + \mu_t\right) \cdot \operatorname{erfc}\left(\frac{\mu_t}{\sqrt{2}\sigma_t} + \frac{\sigma_t}{\sqrt{2}}\right)\right) \cdot \gamma \cdot w^\star \\
&\qquad\qquad - \frac{\eta}{2}\left(\exp\left(\frac{\sigma_t^2}{2} - \mu_t\right) \cdot \operatorname{erfc}\left(-\frac{\mu_t}{\sqrt{2}\sigma_t} + \frac{\sigma_t}{\sqrt{2}}\right) \right. \\
&\qquad\qquad +\exp\left(\frac{\sigma_t^2}{2} + \mu_t\right) \cdot \operatorname{erfc}\left(\frac{\mu_t}{\sqrt{2}\sigma_t} + \frac{\sigma_t}{\sqrt{2}}\right) \\
&\qquad\qquad \left. -\frac{2\sqrt{2}}{\sigma_t\sqrt{\pi}}\exp\left(-\frac{\mu_t^2}{2\sigma_t^2}\right)\right) \cdot (2h_{\text{in}}^t - 2({h_{\text{in}}^t}^\top w^\star)w^\star) \cdot \sigma_{\text{in}}^2 \\
&= h_{\text{in}}^{t} - \frac{\eta}{2} \cdot \alpha_1(\mu_t, \sigma_t) \cdot \gamma \cdot w^\star - \frac{\eta}{2} \cdot \alpha_2(\mu_t, \sigma_t) \cdot (2h_{\text{in}}^t - 2({h_{\text{in}}^t}^\top w^\star)w^\star) \cdot \sigma_{\text{in}}^2 \,. \qquad \text{(E.20)}
\end{aligned}$$

Notice that the update of h_{in}^{t+1} is split into two components, one along $w^\star$ and the other along the orthogonal component $2h_{\text{in}}^t - 2({h_{\text{in}}^t}^\top w^\star)w^\star$. We will now argue that since at initialization, the component along $(I - w^\star {w^\star}^\top)$ is zero then it will remain zero. In particular, we have:

$$ {h_{\text{in}}^0}^\top (I - w^\star {w^\star}^\top) \propto {w^\star}^\top (I - w^\star {w^\star}^\top) = 0 \,. \qquad \text{(E.21)}$$

With (E.20), we can argue that if $(I - w^\star {w^\star}^\top) h_{\text{in}}^t = 0$, then $(I - w^\star {w^\star}^\top)\widetilde{h}_{\text{inv}}^{t+1} = 0$ implying that $(I - w^\star {w^\star}^\top)\widetilde{h}_{\text{in}}^{t} = 0$ for all $t > 0$. Hence, we have:

$$\begin{aligned}
\widetilde{h}_{\text{inv}}^{t+1} &= h_{\text{in}}^{t} - \eta \cdot \frac{\partial \mathcal{L}_{\text{st}}(h^t)}{\partial h_{\text{in}}} \\
&= h_{\text{in}}^{t} - \frac{\eta}{2} \cdot \alpha_1(\mu_t, \sigma_t) \cdot \gamma \cdot w^\star \,. \qquad \text{(E.22)}
\end{aligned}$$

Second, we have the update $\widetilde{h}_{\text{sp}}^{t+1}$ given by:

$$\begin{aligned}
\widetilde{h}_{\text{sp}}^{t+1} &= h_{\text{sp}}^{t} - \eta \cdot \frac{\partial \mathcal{L}_{\text{st}}(h^t)}{\partial h_{\text{sp}}} \\
&= h_{\text{sp}}^{t} - \frac{\eta}{2}\left(\exp\left(\frac{\sigma_t^2}{2} - \mu_t\right) \cdot \operatorname{erfc}\left(-\frac{\mu_t}{\sqrt{2}\sigma_t} + \frac{\sigma_t}{\sqrt{2}}\right) \right. \\
&\qquad\qquad \left. +\exp\left(\frac{\sigma_t^2}{2} + \mu_t\right) \cdot \operatorname{erfc}\left(\frac{\mu_t}{\sqrt{2}\sigma_t} + \frac{\sigma_t}{\sqrt{2}}\right) - \frac{2\sqrt{2}}{\sigma_t\sqrt{\pi}}\exp\left(-\frac{\mu_t^2}{2\sigma_t^2}\right)\right) \cdot h_{\text{sp}}^t \cdot \sigma_{\text{sp}}^2 \\
&= h_{\text{sp}}^{t} - \frac{\eta}{2} \cdot \alpha_2(\mu_t, \sigma_t) \cdot h_{\text{sp}}^t \cdot \sigma_{\text{sp}}^2 \,. \qquad \text{(E.23)}
\end{aligned}$$

Re-writing the expressions (E.22) and (E.23) for the update of $\widetilde{h}^{t+1}$, we have:

$$\widetilde{h}_{\mathrm{in}}^{t+1} = h_{\mathrm{in}}^{t}(1 - \frac{\eta}{2} \cdot \alpha_1(\mu_t, \sigma_t) \cdot \gamma^2/\mu_t)\,. \tag{E.24}$$

$$\widetilde{h}_{\mathrm{sp}}^{t+1} = h_{\mathrm{sp}}^{t}(1 - \frac{\eta}{2} \cdot \alpha_2(\mu_t, \sigma_t) \cdot \sigma_{\mathrm{sp}}^2)\,. \tag{E.25}$$

Here, we replace $h_{\mathrm{sp}}^t = \mu_t \cdot w^\star/\gamma$ in (E.22) to get (E.24). Updates in (E.24) and (E.25) show that $\widetilde{h}_{\mathrm{inv}}^{t+1}$ remains in the direction of h_{in}^t and $\widetilde{h}_{\mathrm{sp}}^{t+1}$ remains in the direction of h_{sp}^t.

Part-2. Now we will derive conditions under which h_{in}^t and h_{sp}^t will show monotonic behavior for necessary and sufficient conditions. We will first argue the condition under which ST will provably fail and converge to a classifier with a random target performance. For this, at every t, if we have:

$$\frac{\left\|\widetilde{h}_{\mathrm{sp}}^{t+1}\right\| 2}{\left\|\widetilde{h}^{t+1}\right\| 2} > \left\|h_{\mathrm{sp}}^t\right\| 2\,, \tag{E.26}$$

then we can argue that as $t \to \infty$, we have $\left\|h_{\mathrm{sp}}^t\right\| 2 = 1$ and hence, the ST classifier will have random target performance. Thus, we will focus on conditions, under which the norm on $\left\|h_{\mathrm{sp}}^t\right\| 2$ increases with t. Re-writing (E.26), we have:

$$\left\|\widetilde{h}_{\mathrm{sp}}^{t+1}\right\| 2 > \left\|\widetilde{h}^{t+1}\right\| 2 \cdot \left\|h_{\mathrm{sp}}^t\right\| 2 \tag{E.27}$$

$$\left\|\widetilde{h}_{\mathrm{sp}}^{t+1}\right\| 2 > \left(\left\|\widetilde{h}_{\mathrm{sp}}^{t+1}\right\| 2 + \left\|\widetilde{h}_{\mathrm{in}}^{t+1}\right\| 2\right) \cdot \left\|h_{\mathrm{sp}}^t\right\| 2 \tag{E.28}$$

$$\left\|\widetilde{h}_{\mathrm{sp}}^{t+1}\right\| 2 \cdot \left(1 - \left\|h_{\mathrm{sp}}^t\right\| 2\right) > \left\|\widetilde{h}_{\mathrm{in}}^{t+1}\right\| 2 \cdot \left\|h_{\mathrm{sp}}^t\right\| 2 \tag{E.29}$$

$$\frac{\left\|\widetilde{h}_{\mathrm{sp}}^{t+1}\right\| 2}{\left\|h_{\mathrm{sp}}^t\right\| 2} > \frac{\left\|\widetilde{h}_{\mathrm{in}}^{t+1}\right\| 2}{\left\|h_{\mathrm{in}}^t\right\| 2}\,. \tag{E.30}$$

Plugging in (E.24) and (E.25) into (E.30), we get:

$$\left|1 - \frac{\eta}{2} \cdot \alpha_2(\mu_t, \sigma_t) \cdot \sigma_{\mathrm{sp}}^2\right| > \left|1 - \frac{\eta}{2} \cdot \alpha_1(\mu_t, \sigma_t) \cdot \gamma^2/\mu_t\right|\,. \tag{E.31}$$

For small enough η, we have the necessary condition for the failure of ST as:

$$\alpha_2(\mu_t, \sigma_t) \cdot \sigma_{\mathrm{sp}}^2 < \alpha_1(\mu_t, \sigma_t) \cdot \gamma^2/\mu_t\,. \tag{E.32}$$

Now we show in Lemma E.5.4 and Lemma E.5.3 that if the conditions assumed in the theorem continue to hold, then we can success and failure respectively.

□

Lemma E.5.3 (Necessary conditions for ST)**.** *Define α_1 and α_2 as in* (E.17) *and* (E.18) *respectively. If $\sigma_{\text{sp}} \geqslant 1$ and $\gamma \leqslant \frac{1}{2\sqrt{\sigma_{\text{sp}}}}$, then we have for all t:*

$$\alpha_2(\mu_t, \sigma_t) \cdot \frac{\sigma_{\text{sp}}^2 \cdot \mu_t}{\gamma^2} \leqslant \alpha_1(\mu_t, \sigma_t)\,. \tag{E.33}$$

Proof. We upper bound and lower bound α_1 and α_2 by using the properties of $\mathrm{r}\,(\cdot)$. Recall:

$$\alpha_1(\mu_t, \sigma_t) = \sqrt{\frac{2}{\pi}}\exp\left(-\frac{\mu_t^2}{2\sigma_t^2}\right)\left[\mathrm{r}\left(\sigma_t + \frac{\mu_t}{\sigma_t}\right) - \mathrm{r}\left(\sigma_t - \frac{\mu_t}{\sigma_t}\right)\right]. \tag{E.34}$$

and

$$\alpha_2(\mu_t, \sigma_t) = \sqrt{\frac{2}{\pi}}\exp\left(-\frac{\mu_t^2}{2\sigma_t^2}\right)\left[\mathrm{r}\left(\sigma_t + \frac{\mu_t}{\sigma_t}\right) + \mathrm{r}\left(\sigma_t - \frac{\mu_t}{\sigma_t}\right) - \frac{2}{\sigma_t}\right]. \tag{E.35}$$

We now use Taylor's expansion on $\mathrm{r}\,(\cdot)$ and we get:

$$\mathrm{r}\,(\sigma_t) + \mathrm{r}'\,(\sigma_t) \cdot \left(\frac{\mu_t}{\sigma_t}\right) \leqslant \mathrm{r}\left(\sigma_t + \frac{\mu_t}{\sigma_t}\right) \leqslant \mathrm{r}\,(\sigma_t) + \mathrm{r}'\,(\sigma_t) \cdot \left(\frac{\mu_t}{\sigma_t}\right) + R''\left(\frac{\mu_t}{\sigma_t}\right)^2 \tag{E.36}$$

and similarly, we get:

$$\mathrm{r}\,(\sigma_t) - \mathrm{r}'\,(\sigma_t) \cdot \left(\frac{\mu_t}{\sigma_t}\right) \leqslant \mathrm{r}\left(\sigma_t - \frac{\mu_t}{\sigma_t}\right) \leqslant \mathrm{r}\,(\sigma_t) - \mathrm{r}'\,(\sigma_t) \cdot \left(\frac{\mu_t}{\sigma_t}\right) + R''\left(\frac{\mu_t}{\sigma_t}\right)^2 \tag{E.37}$$

where $R'' = \mathrm{r}''\,(\sigma_0)$. This is because $\mathrm{r}''\,(\cdot)$ takes positive values and is a decreasing function in σ_t (refer to Lemma E.7.2). We now lower bound $\alpha_1(\mu_t, \sigma_t)$ and upper bound $\alpha_2(\mu_t, \sigma_t)$:

$$\frac{\alpha_1(\mu_t, \sigma_t)}{\sqrt{\frac{2}{\pi}}\exp\left(-\frac{\mu_t^2}{2\sigma_t^2}\right)} \geqslant 2\mathrm{r}'\,(\sigma_t) \cdot \left(\frac{\mu_t}{\sigma_t}\right) - R''\left(\frac{\mu_t}{\sigma_t}\right)^2 \tag{E.38}$$

$$\frac{\alpha_2(\mu_t, \sigma_t)}{\sqrt{\frac{2}{\pi}}\exp\left(-\frac{\mu_t^2}{2\sigma_t^2}\right)} \leqslant 2\mathrm{r}\,(\sigma_t) + 2 \cdot R''\left(\frac{\mu_t}{\sigma_t}\right)^2 \tag{E.39}$$

Substituting the lower bound and upper bound in (E.33) gives us the following as stricter a necessary condition (i.e., (E.40) implies (E.33)):

$$\left[2\mathrm{r}\,(\sigma_t) + 2 \cdot R''\left(\frac{\mu_t}{\sigma_t}\right)^2 - \frac{2}{\sigma_t}\right] \cdot \frac{\sigma_{\text{sp}}^2 \cdot \mu_t}{\gamma^2} \leqslant 2\mathrm{r}'\,(\sigma_t) \cdot \left(\frac{\mu_t}{\sigma_t}\right) - R''\left(\frac{\mu_t}{\sigma_t}\right)^2 \tag{E.40}$$

$$\iff \left[2\mathrm{r}\left(\sigma_t\right) + 2 \cdot R'' \left(\frac{\mu_t}{\sigma_t}\right)^2 - \frac{2}{\sigma_t}\right] \cdot \frac{\sigma_{\mathrm{sp}}^2}{\gamma^2} \leqslant 2\mathrm{r}'\left(\sigma_t\right) \cdot \left(\frac{1}{\sigma_t}\right) - R''\left(\frac{\mu_t}{\sigma_t^2}\right) \tag{E.41}$$

$$\iff \left[\mathrm{r}\left(\sigma_t\right) + R'' \left(\frac{\mu_t}{\sigma_t}\right)^2 - \frac{1}{\sigma_t}\right] \cdot \frac{\sigma_{\mathrm{sp}}^2}{\gamma^2} \leqslant \mathrm{r}\left(\sigma_t\right) - \frac{1}{\sigma_t} - \frac{R''}{2}\left(\frac{\mu_t}{\sigma_t^2}\right) \tag{E.42}$$

$$\iff \left[R'' \left(\frac{\mu_t}{\sigma_t}\right)^2\right] \cdot \frac{\sigma_{\mathrm{sp}}^2}{\gamma^2} + \frac{R''}{2}\left(\frac{\mu_t}{\sigma_t^2}\right) \leqslant \left(\mathrm{r}\left(\sigma_t\right) - \frac{1}{\sigma_t}\right) \cdot \left(1 - \frac{\sigma_{\mathrm{sp}}^2}{\gamma^2}\right) \tag{E.43}$$

$$\iff \left[R'' \left(\frac{\mu_t^2}{\sigma_t}\right)\right] \cdot \frac{\sigma_{\mathrm{sp}}^2}{\gamma^2} + \frac{R''}{2}\left(\frac{\mu_t}{\sigma_t}\right) \leqslant \left(\sigma_t \mathrm{r}\left(\sigma_t\right) - 1\right) \cdot \left(1 - \frac{\sigma_{\mathrm{sp}}^2}{\gamma^2}\right) \tag{E.44}$$

Now, we will argue the monotonicity of LHS and RHS in (E.44). Observe that LHS is increasing in μ_t and decreasing in σ_t and RHS is decreasing in σ_t as $(\sigma_t \mathrm{r}\left(\sigma_t\right) - 1)$ is increasing (and the multiplier is negative). Moreover, if (E.44) holds true for maximum value of RHS and minimum of LHS, then we would have (E.33). Thus substituting $\mu_t = \gamma$ and $\sigma_t = \sigma_0$ in LHS and $\sigma_t = \sigma_{\mathrm{sp}}$ in RHS, we get:

$$\left[R'' \left(\frac{\gamma^2}{\sigma_0}\right)\right] \cdot \frac{\sigma_{\mathrm{sp}}^2}{\gamma^2} + \frac{R''}{2}\left(\frac{\gamma}{\sigma_0}\right) \leqslant \left(\sigma_{\mathrm{sp}} \mathrm{r}\left(\sigma_{\mathrm{sp}}\right) - 1\right) \cdot \left(1 - \frac{\sigma_{\mathrm{sp}}^2}{\gamma^2}\right) \tag{E.45}$$

$$\iff R'' \cdot \frac{\sigma_{\mathrm{sp}}^2}{\sigma_0} + \frac{R''}{2}\left(\frac{\gamma}{\sigma_0}\right) \leqslant \left(\sigma_{\mathrm{sp}} \mathrm{r}\left(\sigma_{\mathrm{sp}}\right) - 1\right) \cdot \left(1 - \frac{\sigma_{\mathrm{sp}}^2}{\gamma^2}\right) \tag{E.46}$$

$$\tag{E.47}$$

Taking $\gamma \leqslant \frac{1}{2\sqrt{\sigma_{\mathrm{sp}}}}$ and substituting $R'' = \mathrm{r}''\left(\sigma_0\right)$:

$$(5/4) \cdot \mathrm{r}''\left(\sigma_0\right) \cdot \sigma_{\mathrm{sp}} \leqslant \left(\sigma_{\mathrm{sp}} \mathrm{r}\left(\sigma_{\mathrm{sp}}\right) - 1\right) \cdot \left(1 - 4 \cdot \sigma_{\mathrm{sp}}^3\right) \tag{E.48}$$

Analytically solving the above expression, we get that (E.48) is satisfied for all values of $\sigma_{\mathrm{sp}} \geqslant 1$ when $d_{\mathrm{sp}} \geqslant 1$. For example, the expression in (E.48) is also satisfied for the problem parameter used in the running example of the main paper.

□

As a remark, we note that in the proof of Lemma E.5.3, the conditions derived are loose because of the relaxations made to simply the proof. In principle, the proof (and hence the conditions) can be tightened by carefully propagating second-order terms (which depend on σ_t) in (E.37).

Lemma E.5.4 (Sufficiency conditions for ST)**.** *Define α_1 and α_2 as in* (E.17) *and* (E.18) *respectively. If $\sigma_{\mathrm{sp}} \leqslant \gamma$, then we have for all t:*

$$\alpha_2(\mu_t, \sigma_t) \cdot \frac{\sigma_{\mathrm{sp}}^2 \cdot \mu_t}{\gamma^2} \geqslant \alpha_1(\mu_t, \sigma_t)\,. \tag{E.49}$$

Proof. We upper bound and lower bound α_1 and α_2 by using the properties of $\mathrm{r}(\cdot)$. Recall:

$$\alpha_1(\mu_t, \sigma_t) = \sqrt{\frac{2}{\pi}} \exp\left(-\frac{\mu_t^2}{2\sigma_t^2}\right)\left[\mathrm{r}\left(\sigma_t + \frac{\mu_t}{\sigma_t}\right) - \mathrm{r}\left(\sigma_t - \frac{\mu_t}{\sigma_t}\right)\right]. \tag{E.50}$$

and

$$\alpha_2(\mu_t, \sigma_t) = \sqrt{\frac{2}{\pi}} \exp\left(-\frac{\mu_t^2}{2\sigma_t^2}\right)\left[\mathrm{r}\left(\sigma_t + \frac{\mu_t}{\sigma_t}\right) + \mathrm{r}\left(\sigma_t - \frac{\mu_t}{\sigma_t}\right) - \frac{2}{\sigma_t}\right]. \tag{E.51}$$

We now use Taylor's expansion on $\mathrm{r}(\cdot)$ and we get:

$$\mathrm{r}(\sigma_t) + \mathrm{r}'(\sigma_t)\cdot\left(\frac{\mu_t}{\sigma_t}\right) \leqslant \mathrm{r}\left(\sigma_t + \frac{\mu_t}{\sigma_t}\right) \leqslant \mathrm{r}(\sigma_t) + \mathrm{r}'(\sigma_t)\cdot\left(\frac{\mu_t}{\sigma_t}\right) + \mathrm{r}''(\sigma_t)\cdot\left(\frac{\mu_t}{\sigma_t}\right)^2 \tag{E.52}$$

and similarly, we get:

$$\mathrm{r}(\sigma_t) - \mathrm{r}'(\sigma_t)\cdot\left(\frac{\mu_t}{\sigma_t}\right) + \mathrm{r}''(\sigma_t)\cdot\left(\frac{\mu_t}{\sigma_t}\right)^2 \leqslant \mathrm{r}\left(\sigma_t - \frac{\mu_t}{\sigma_t}\right) \leqslant \mathrm{r}(\sigma_t) - \mathrm{r}'(\sigma_t)\cdot\left(\frac{\mu_t}{\sigma_t}\right) + R''\left(\frac{\mu_t}{\sigma_t}\right)^2 \tag{E.53}$$

where $R'' = \mathrm{r}''(\sigma_0)$. This is because $\mathrm{r}''(\cdot)$ takes positive values and is a decreasing function in σ_t (refer to Lemma E.7.2). We now lower bound $\alpha_1(\mu_t, \sigma_t)$ and upper bound $\alpha_2(\mu_t, \sigma_t)$:

$$\frac{\alpha_1(\mu_t, \sigma_t)}{\sqrt{\frac{2}{\pi}}\exp\left(-\frac{\mu_t^2}{2\sigma_t^2}\right)} \leqslant 2\mathrm{r}'(\sigma_t)\cdot\left(\frac{\mu_t}{\sigma_t}\right) \tag{E.54}$$

$$\frac{\alpha_2(\mu_t, \sigma_t)}{\sqrt{\frac{2}{\pi}}\exp\left(-\frac{\mu_t^2}{2\sigma_t^2}\right)} \geqslant 2\mathrm{r}(\sigma_t) + \mathrm{r}''(\sigma_t)\cdot\left(\frac{\mu_t}{\sigma_t}\right)^2 - \frac{2}{\sigma_t} \tag{E.55}$$

Substituting the lower bound and upper bound in (E.49) gives us the following as stricter a sufficient condition (i.e., (E.56) implies (E.49)):

$$\left[2\mathrm{r}(\sigma_t) + \mathrm{r}''(\sigma_t)\cdot\left(\frac{\mu_t}{\sigma_t}\right)^2 - \frac{2}{\sigma_t}\right]\cdot\frac{\sigma_{\mathrm{sp}}^2\cdot\mu_t}{\gamma^2} \geqslant 2\mathrm{r}'(\sigma_t)\cdot\left(\frac{\mu_t}{\sigma_t}\right) \tag{E.56}$$

$$\iff \left[2\mathrm{r}(\sigma_t) + \mathrm{r}''(\sigma_t)\cdot\left(\frac{\mu_t}{\sigma_t}\right)^2 - \frac{2}{\sigma_t}\right] \geqslant 2\mathrm{r}'(\sigma_t)\cdot\left(\frac{\mu_t}{\sigma_t}\right)\cdot\frac{\gamma^2}{\sigma_{\mathrm{sp}}^2\cdot\mu_t} \tag{E.57}$$

$$\iff 2\mathrm{r}(\sigma_t) + \mathrm{r}''(\sigma_t)\cdot\left(\frac{\mu_t}{\sigma_t}\right)^2 - \frac{2}{\sigma_t} - 2\mathrm{r}'(\sigma_t)\cdot\left(\frac{\mu_t}{\sigma_t}\right)\cdot\frac{\gamma^2}{\sigma_{\mathrm{sp}}^2\cdot\mu_t} \geqslant 0 \tag{E.58}$$

$$\iff 2\mathrm{r}(\sigma_t)\cdot\sigma_t + \mathrm{r}''(\sigma_t)\cdot\frac{\mu_t^2}{\sigma_t} - 2 - 2\mathrm{r}'(\sigma_t)\cdot\frac{\gamma^2}{\sigma_{\mathrm{sp}}^2} \geqslant 0 \tag{E.59}$$

$$\iff 2\mathrm{r}'\left(\sigma_t\right) + \mathrm{r}''\left(\sigma_t\right) \cdot \frac{\mu_t^2}{\sigma_t} - 2\mathrm{r}'\left(\sigma_t\right) \cdot \frac{\gamma^2}{\sigma_{\mathrm{sp}}^2} \geqslant 0 \tag{E.60}$$

$$\iff \mathrm{r}''\left(\sigma_t\right) \cdot \frac{\mu_t^2}{\sigma_t} + 2\mathrm{r}'\left(\sigma_t\right) \cdot \left[1 - \frac{\gamma^2}{\sigma_{\mathrm{sp}}^2}\right] \geqslant 0 \tag{E.61}$$

Hence, when $\left[1 - \frac{\gamma^2}{\sigma_{\mathrm{sp}}^2}\right] \leqslant 0$, we have condition in (E.61) hold true as $\mathrm{r}'\left(\sigma_t\right)$ is always negative. Hence, the condition $\gamma \geqslant \sigma_{\mathrm{sp}}$ gives us the necessary condition. □

Proof of Proposition 6.4.3

For convenience, we first restate the Proposition 6.4.3 which gives us a closed form solution for (6.3) when $\rho = 0$. Then, we provide the proof, focusing first on the case of $k = 1$, and then showing that extension to $k > 1$ is straightforward and renders the final form in the proposition that follows.

Proposition E.5.5 (Barlow Twins solution)**.** *The solution for* (6.3) *is* $U_k^\top \Sigma_{\mathsf{A}}^{-1/2}$ *where* U_k *are the top* k *eigenvectors of* $\Sigma_{\mathsf{A}}^{-1/2}\, \widetilde{\Sigma}\, \Sigma_{\mathsf{A}}^{-1/2}$*. Here,* $\Sigma_{\mathsf{A}} := \mathbb{E}_{a \sim \mathrm{P}_{\mathsf{A}}}[aa^\top]$ *is the covariance over augmentations, and* $\widetilde{\Sigma} := \mathbb{E}_{x \sim \mathrm{P}_{\mathsf{U}}}[\widetilde{a}(x)\widetilde{a}(x)^\top]$ *is the covariance matrix of mean augmentations* $\widetilde{a}(x) := \mathbb{E}_{\mathrm{P}_{\mathsf{A}}(a|x)}[a]$*.*

Proof. We will use $\phi(x)$ to denote $\phi^\top x$ where $\phi \in \mathbb{R}^d$. Throughout the proof, we use a to denote augmentation and x to denote the input. We will use $\mathrm{P}_{\mathsf{A}}(a \mid x)$ as the probability measure over the space of augmentations $\mathcal{A}$, given some input $x \in \mathcal{X}$ (with corresponding density) $p_{\mathsf{A}}(\cdot \mid x)$. Next, we use $p_{\mathsf{A}}(\cdot)$ to denote the density associate with the marginal probability measure over augmentations: $\mathrm{P}_{\mathsf{A}} = \int_{\mathcal{X}} \mathrm{P}_{\mathsf{A}}(a \mid x) \mathrm{dP}_{\mathsf{U}}$. Finally, the joint distribution over positive pairs $A_+(a_1, a_2) = \int_{\mathcal{X}} \mathrm{P}_{\mathsf{A}}(a_1 \mid x)\mathrm{P}_{\mathsf{A}}(a_2 \mid x)\mathrm{dP}_{\mathsf{U}}$, gives us the positive pair graph over augmentations.

Before we solve the optimization problem in (6.3) for $\Phi \in \mathbb{R}^{k \times d}$ for any general k, let us first consider the case where $k = 1$, *i.e.* we only want to find a single linear projection ϕ. The constraint $\rho = 0$, transfers onto ϕ in the following way:

$$\mathbb{E}_{a \sim \mathrm{P}_{\mathsf{A}}}[\phi(a)^2] = 1 \quad \equiv \quad \phi^\top \Sigma_A \phi = 1 \tag{E.62}$$

Under the above constraint we want to minimize the invariance loss, which according to Lemma E.7.3 is given by $2 \cdot \int_{\mathcal{A}} \phi(a) L(\phi)(a)\ \mathrm{dP}_{\mathsf{A}}$, where $L(\phi)(\cdot)$ is the following linear operator.

$$L(\phi)(a) = \phi(a) - \int_{\mathcal{A}} \frac{A_+(a, a')}{p_{\mathsf{A}}(a)} \cdot \phi(a')\ \mathrm{d}a'. \tag{E.63}$$

Based on the definition of the operator, we can reformulate the constrained optimization for contrastive pretraining as:

$$\begin{aligned}
&\operatorname*{arg\,min}_{\phi:\phi^\top \Sigma_A \phi = 1} \int_{\mathcal{A}} \phi(a) \cdot L(\phi)(a) \; \mathrm{dP}_{\mathsf{A}} && \text{(E.64)} \\
\implies &\operatorname*{arg\,min}_{\phi:\phi^\top \Sigma_A \phi = 1} \; \mathbb{E}_{a \sim \mathrm{P}_{\mathsf{A}}}[\phi(a)^2] - \int_{\mathcal{A}} \int_{\mathcal{A}} \phi(a) \cdot \phi(a') \cdot A_+(a, a') \; \mathrm{d}a \mathrm{d}a' && \text{(E.65)} \\
\implies &\operatorname*{arg\,min}_{\phi:\phi^\top \Sigma_A \phi = 1} \; \mathbb{E}_{a \sim \mathrm{P}_{\mathsf{A}}}[\phi(a)^2] - \int_{\mathcal{X}} \int_{\mathcal{A}} \int_{\mathcal{A}} p_{\mathsf{A}}(a \mid x) p_{\mathsf{A}}(a' \mid x) \cdot \phi(a)\phi(a') \; \mathrm{dP}_{\mathsf{U}} && \text{(E.66)} \\
\implies &\operatorname*{arg\,min}_{\phi:\phi^\top \Sigma_A \phi = 1} \; \mathbb{E}_{a \sim \mathrm{P}_{\mathsf{A}}}[\phi(a)^2] - \int_{\mathcal{X}} [\widetilde{\phi}(x)]^2 \; \mathrm{dP}_{\mathsf{U}}, && \text{(E.67)}
\end{aligned}$$

where $\widetilde{\phi}(x) = \mathbb{E}_{a \sim \mathrm{P}_{\mathsf{A}}(\cdot|x)} \phi(x) = \mathbb{E}_{c \sim \mathrm{Unif}[0,1]^d}[\phi^\top (c \odot x)]$. Note that,

$$\begin{aligned}
\widetilde{\phi}(x)^2 &= \left(\mathbb{E}_{c \sim \mathrm{Unif}[0,1]^d}[\phi^\top (c \odot x)]\right)^2 && \text{(E.68)} \\
&= \phi^\top (\mathbb{E}_{c \sim \mathrm{Unif}[0,1]^d}[c \odot x]) (\mathbb{E}_{c \sim \mathrm{Unif}[0,1]^d}[c \odot x])^\top \phi && \text{(E.69)} \\
\implies \int_{\mathcal{X}} [\widetilde{\phi}(x)]^2 \; \mathrm{dP}_{\mathsf{U}} &= \phi^\top \widetilde{\Sigma} \phi && \text{(E.70)}
\end{aligned}$$

Further, since $\mathbb{E}_{a \sim \mathrm{P}_{\mathsf{A}}}[\phi(a)^2] = \phi^\top \Sigma \phi$ we can now rewrite our main optimization problem for $k = 1$ as:

$$\begin{aligned}
&\operatorname*{arg\,min}_{\phi:\phi^\top \Sigma_A \phi = 1} \; \phi^\top \Sigma_A \phi - \phi^\top \widetilde{\Sigma} \phi && \text{(E.71)} \\
&= \operatorname*{arg\,max}_{\phi:\phi^\top \Sigma_A \phi = 1} \phi^\top \widetilde{\Sigma} \phi && \text{(E.72)}
\end{aligned}$$

Recall that in our setup both $\widetilde{\Sigma}$ and Σ_A are positive definite and invertible matrices. To solve the above problem, let's consider a re-parameterization: $\phi' = \Sigma_A^{1/2} \phi$, thus $\phi^\top \Sigma_A \phi = 1$, is equivalent to the constraint $\|\phi'\|_2^2 = 1$. Based on this re-parameterization we are now solving:

$$\operatorname*{arg\,max}_{\|\phi'\|_2^2 = 1} \; \phi'^\top \Sigma_A^{-1/2} \cdot \widetilde{\Sigma} \cdot \Sigma_A^{-1/2} \phi', \qquad \text{(E.73)}$$

which is nothing but the top eigenvector for $\Sigma_A^{-1/2} \cdot \widetilde{\Sigma} \cdot \Sigma_A^{-1/2}$.

Now, to extend the above argument from $k = 1$ to $k > 1$, we need to care of one additional form of constraint in the form of feature diversity: $\phi_i^\top \Sigma_A \phi_j = 0$ when $i \neq j$. But, we can easily redo the reformulations above and arrive at the following optimization problem:

$$\operatorname*{arg\,max}_{\substack{\|\phi_i'\|_2^2 = 1, \;\; \forall i \\ \phi_i'^\top \phi_j' = 0, \;\; \forall i \neq j}} \quad \left[\phi_1', \phi_2', \ldots, \phi_k'\right]^\top \Sigma_A^{-1/2} \cdot \widetilde{\Sigma} \cdot \Sigma_A^{-1/2} \left[\phi_1', \phi_2', \ldots, \phi_k'\right], \qquad \text{(E.74)}$$

where $\phi_i' = \Sigma_A^{1/2}\phi_i$. The above is nothing but the top k eigenvectors for the matrix $\Sigma_A^{-1/2} \cdot \widetilde{\Sigma} \cdot \Sigma_A^{-1/2}$. This completes the proof of Proposition E.5.5. □

Analysis with $\rho > 0$ in Contrastive Pretraining Objective (6.3)

In (6.3) we considered the strict version of the optimization problem where $\rho = 0$. Here, we will consider the following optimization problem that we optimize for our experiments in the simplified setup:

$$\mathcal{L}_{\text{cl}}(\Phi, \kappa) \;:=\; \mathbb{E}_{x\sim \mathrm{P_U}}\mathbb{E}_{a_1,a_2\sim \mathrm{P_A}(\cdot|x)}\;\|\Phi(a_1) - \Phi(a_2)\|_2^2 + \kappa \cdot \left\|\mathbb{E}_{a\sim \mathrm{P_A}}\left[\Phi(a)\Phi(a)^\top\right] - \mathbf{I}_k\right\| F^2, \tag{E.75}$$

where $\kappa > 0$ is some finite constant (note that every ρ corresponds to some κ and particularly $\rho = 0$, corresponds to $\kappa = \infty$). Let $\Phi^\star$ be the solution for (6.3) with $\rho = 0$, *i.e.* the solution described in Proposition 6.4.3. Now, we will show that in practice we can provably recover something close to $\Phi^\star$ when κ is large enough.

Theorem E.5.6 (Solution for (E.75) is approximately equal to $\Phi^\star$)**.** *If $\widehat{\Phi}$ is some solution that achieves low values of the objective $\mathcal{L}_{\text{cl}}(\Phi, \kappa)$ in* (E.75), i.e., *$\mathcal{L}_{\text{cl}}(\widehat{\Phi}, \kappa) \leqslant \epsilon$, then there exists matrix $W \in \mathbb{R}^{k\times k}$ such that:*

$$\mathbb{E}_{a\sim \mathrm{P_A}}\|W\cdot\Phi^\star(a) - \widehat{\Phi}(a)\|_2^2 \leqslant \frac{k\epsilon}{2\gamma_{k+1}},$$
$$\textit{where,}\;\; \gamma_{k+1} \geqslant \frac{2\gamma_1^2}{k\epsilon}\cdot\left(1 - \sqrt{\frac{\epsilon}{\kappa}}\right) - \frac{\gamma_1}{k},$$

where γ_{k+1} is the the $k+1^{th}$ eigenvalue for $\mathbf{I}_d - \Sigma_A^{-1/2}\,\widetilde{\Sigma}\,\Sigma_A^{-1/2}$. Here, $\lambda_1 \leqslant \lambda_2 \leqslant \ldots \leqslant \lambda_d$.

Proof. Since we know that $\mathcal{L}_{\text{cl}}(\widehat{\Phi}, \kappa) \leqslant \epsilon$, we can individually bound the invariance loss and the regularization term:

$$\mathbb{E}_{x\sim \mathrm{P_U}}\mathbb{E}_{a_1,a_2\sim \mathrm{P_A}(\cdot|x)}\;\|\widehat{\Phi}(a_1) - \widehat{\Phi}(a_2)\|_2^2 \leqslant \epsilon \tag{E.76}$$

$$\left\|\mathbb{E}_{a\sim \mathrm{P_A}}\left[\widehat{\Phi}(a)\widehat{\Phi}(a)^\top\right] - \mathbf{I}_k\right\| F^2 \leqslant \frac{\epsilon}{\kappa} \tag{E.77}$$

Thus,

$$\forall i \in [k]:\;\; 1 - \sqrt{\frac{\epsilon}{\kappa}} \leqslant \widehat{\phi}_i^\top\;\Sigma_A\;\widehat{\phi}_i \leqslant 1 + \sqrt{\frac{\epsilon}{\kappa}} \tag{E.78}$$

$$\forall i \in [k]:\;\; \mathbb{E}_{x\sim \mathrm{P_U}}\mathbb{E}_{a_1,a_2\sim \mathrm{P_A}(\cdot|x)}(\widehat{\phi}_i^\top a_1 - \widehat{\phi}_i^\top a_2)^2 \leqslant \epsilon \tag{E.79}$$

Let $\phi_1^\star, \phi_2^\star, \phi_3^\star, \ldots, \phi_d^\star$ be the solution returned by the analytical solution for $\rho = 0$, *i.e.* the solution in Proposition 6.4.3. Now, since $\Phi^\star$ would span $\mathbb{R}^d$ when Σ_A is full rank, we can denote:

$$\widehat{\phi}_i = \sum_{j=1}^{d} \eta_i^{(j)} \phi_j^\star \tag{E.80}$$

Now from Lemma E.7.3, the invariance loss for $\widehat{\phi}_i$ can be written using the operator $L(\phi)(a) = \phi(a) - \int_{\mathcal{A}} \frac{A_+(a,a')}{p_{\mathsf{A}}(a)} \phi(a')\,\mathrm{d}a'$:

$$\begin{aligned}
\text{Invariance Loss}(\widehat{\phi}_i) &:= \mathbb{E}_{x\sim \mathrm{P}_{\mathsf{U}}} \mathbb{E}_{a_1,a_2\sim \mathrm{P}_{\mathsf{A}}(\cdot|x)} (\widehat{\phi}_i^\top a_1 - \widehat{\phi}_i^\top a_2)^2 && \text{(E.81)}\\
&= 2 \cdot \mathbb{E}_{a\sim \mathrm{P}_{\mathsf{A}}} [\widehat{\phi}_i(a) L(\widehat{\phi}_i)(a)] && \text{(E.82)}\\
&= 2 \cdot \mathbb{E}_{a\sim \mathrm{P}_{\mathsf{A}}} \left[\left(\sum_{j=1}^{d} \eta_i^{(j)} \phi_i^\star \right) L \left(\sum_{j=1}^{d} \eta_i^{(j)} \phi_j^\star \right)(a) \right] && \text{(E.83)}\\
&= 2 \cdot \mathbb{E}_{a\sim \mathrm{P}_{\mathsf{A}}} \left[\left(\sum_{j=1}^{d} \eta_i^{(j)} \phi_j^\star \right) \left(\sum_{j=1}^{d} \eta_i^{(j)} L(\phi_j^\star)(a) \right) \right] && \text{(E.84)}\\
&= 2 \cdot \sum_{j=1}^{d} \left(\eta_i^{(j)} \right)^2 \mathbb{E}_{a\sim \mathrm{P}_{\mathsf{A}}} \left[\phi_j^\star(a) L(\phi_j^\star)(a) \right] && \text{(E.85)}\\
&\quad + 2 \cdot \sum_{m=1,n=1,m\neq n}^{d} \eta_i^{(m)} \eta_i^{(n)} \mathbb{E}_{a\sim \mathrm{P}_{\mathsf{A}}} \left[\phi_m^\star(a) L(\phi_n^\star)(a) \right] && \text{(E.86)}
\end{aligned}$$

Since, $\phi_i^\star(\cdot)$ are eigenfunctions of the operator L (HaoChen and Ma, 2022), we can conclude that:

$$\sum_{m=1,n=1,m\neq n}^{d} \eta_i^{(m)} \eta_i^{(n)} \mathbb{E}_{a\sim \mathrm{P}_{\mathsf{A}}} \left[\phi_m^\star(a) L(\phi_n^\star)(a) \right] = 0,$$

and if $\gamma_1 \leqslant \gamma_2 \leqslant \gamma_3 \ldots \leqslant \gamma_d$ are the eigenvalues for $\phi_1^\star, \phi_2^\star, \phi_3^\star, \ldots, \phi_d^\star$ under the decomposition of $L(\phi)(\cdot)$ then:

$$\mathbb{E}_{x\sim \mathrm{P}_{\mathsf{U}}} \mathbb{E}_{a_1,a_2\sim \mathrm{P}_{\mathsf{A}}(\cdot|x)} (\widehat{\phi}_i^\top a_1 - \widehat{\phi}_i^\top a_2)^2 = 2 \cdot \sum_{j=1}^{d} \gamma_j \left(\eta_i^{(j)} \right)^2 \tag{E.87}$$

Recall, we are also aware of a condition on the regularization term: $1 - \sqrt{\frac{\epsilon}{\kappa}} \leqslant \widehat{\phi}_i^\top \, \Sigma_A \, \widehat{\phi}_i \leqslant 1 + \sqrt{\frac{\epsilon}{\kappa}}$.

$$\widehat{\phi}_i^\top \, \Sigma_A \, \widehat{\phi}_i = \left(\sum_{j=1}^{d} \eta_i^{(j)} \phi_j^\star \right)^\top \, \Sigma_A \, \left(\sum_{j=1}^{d} \eta_i^{(j)} \phi_j^\star \right) = \sum_{j=1}^{d} \left(\eta_i^{(j)} \right)^2 \tag{E.88}$$

$$\implies 1 - \sqrt{\frac{\epsilon}{\kappa}} \leqslant \sum_{j=1}^{d} \left(\eta_i^{(j)}\right)^2 \leqslant 1 + \sqrt{\frac{\epsilon}{\kappa}} \;\; \forall i. \tag{E.89}$$

In order to show that the projection of $\widehat{\phi}_i$ on Φ^* is significant, we need to argue that the term $\sum_{j=k+1}^{d} \left(\eta_i^{(j)}\right)^2$ is small. The argument for this begins with the condition on invariance loss, and the fact that $\gamma_1 \leqslant \gamma_2 \leqslant \ldots \leqslant \gamma_k \leqslant \gamma_{k+1} \leqslant \ldots \leqslant \gamma_d$:

$$\frac{\epsilon}{2} \geqslant \sum_{j=k+1}^{d} \left(\eta_i^{(j)}\right)^2 \gamma_j \geqslant \gamma_{k+1} \cdot \left(\sum_{j=k+1}^{d} \left(\eta_i^{(j)}\right)^2\right) \tag{E.90}$$

$$\implies \sum_{j=k+1}^{d} \left(\eta_i^{(j)}\right)^2 \leqslant \frac{\epsilon}{2\gamma_{k+1}} \tag{E.91}$$

Extending the above result $\forall i$ by simply adding the bounds completes the claim of our first result in Theorem E.5.6. Next, we will lower bound the eigenvalue γ_{k+1}. Recall that, $\sum_{j=1}^{k} \left(\eta_i^{(j)}\right)^2 \geqslant 1 - \sqrt{\frac{\epsilon}{\kappa}} - \frac{\epsilon}{2\gamma_{k+1}}$. Thus,

$$\gamma_1 \cdot \left(1 - \sqrt{\frac{\epsilon}{\kappa}} - \frac{\epsilon}{2\gamma_{k+1}}\right) \leqslant \sum_{j=1}^{k} \gamma_j \left(\eta_i^{(j)}\right)^2 \leqslant k\gamma_{k+1} \cdot \frac{\epsilon}{2\gamma_1} \tag{E.92}$$

We assume that all eigenvalues are strictly positive, which is true under our augmentation distribution. Given, $\gamma_{k+1} \geqslant \gamma_1$, we can rearrange the above to get:

$$\gamma_{k+1} \geqslant \frac{2\gamma_1^2}{k\epsilon} \cdot \left(1 - \sqrt{\frac{\epsilon}{\kappa}}\right) - \frac{\gamma_1}{k} \tag{E.93}$$

This completes the claim of our second result in Theorem E.5.6. □

Proof of Theorem 6.4.4

In this section, we prove our main theorem about the recovery of both spurious w_{sp}, invariant w_{in} features by the contrastive learning feature backbone, and also the amplification of the invariant over the spurious feature (where amplification is defined relatively with respect to what is observed in the data distribution alone). We begin by defining some quantities needed for analysis, that are fully determined by the choice of problem parameters for the model in (E.3).

From Section 12.4, we recall the definitions of $w_{\text{in}} := [w^\star, 0, \dots, 0]$ and $w_{\text{sp}} := [0, \dots 0, w']$ where $w' = \mathbf{1}_{d_{\text{sp}}}/\sqrt{d_{\text{sp}}}$. Let us now define u_1, u_2 as the top two eigenvectors of Σ_A with eigenvalues $\lambda_1, \lambda_2 > 0$, (note that in our problem setup both Σ_A and $\widetilde{\Sigma}$ are full rank positive definite matrices), and $\tau := \sqrt{\lambda_1/\lambda_2}$. Next we define α as the angle between u_1 and w_{in}, *i.e.*, $\cos(\alpha) = u_1^\top w_{\text{in}}$. Based on the definitions of α and τ, both of which are fully determined by the eigen decomposition of the post-augmentation feature covariance matrix Σ_A, we now restate Theorem 6.4.4:

Theorem E.5.7 (Formal; CL recovers both invariant w_{in} and spurious w_{sp} but amplifies w_{in})**.** *Under Assumption 1 ($w^\star = {}^{\mathbf{1}_{d_{\text{in}}}}/\sqrt{d_{\text{in}}}$), the CL solution Φ_{cl}=$[\phi_1, \phi_2, ..., \phi_k]$ satisfies $\phi_j^\top w_{\text{in}} = \phi_j^\top w_{\text{sp}} = 0 \; \forall j \geqslant 3$. For τ, α as defined above, the solution for ϕ_1, ϕ_2 is:*

$$\begin{bmatrix} w^\star \cdot \cot(\alpha)/\tau, & w^\star \\ w' \cdot 1/\tau, & w' \cdot \cot(\alpha) \end{bmatrix} \quad \cdot \quad \begin{bmatrix} \cos\theta, & \sin\theta \\ \sin\theta, & -\cos\theta \end{bmatrix},$$

where $0 \leqslant \alpha, \theta \leqslant \pi/2$. Let us redefine $\phi_1 = c_1 w_{\text{in}} + c_3 w_{\text{sp}}$ and $\phi_2 = c_2 w_{\text{in}} + c_4 w_{\text{sp}}$.

For constants $K_1, K_2 > 0$, $\gamma = {}^{K_1 K_2}/_{\sigma_{\text{sp}}}$, $d_{\text{sp}} = {}^{\sigma_{\text{sp}}^2}/_{K_2^2}$, $\forall \epsilon > 0$, $\exists \sigma_{\text{sp}_0}$, such that for $\sigma_{\text{sp}} \geqslant \sigma_{\text{sp}_0}$:

$$\frac{K_1 K_2^2 d_{\text{in}}}{2L\sigma_{\text{in}}^2(d_{\text{in}} - 1)} + \epsilon \;\geqslant\; \frac{c_1}{c_3} \;\geqslant\; \frac{K_1 K_2^2 d_{\text{in}}}{2L\sigma_{\text{in}}^2(d_{\text{in}} - 1)} - \epsilon$$
$$\frac{L\sqrt{d_{\text{sp}}}}{\gamma} + \epsilon \;\geqslant\; \left|\frac{c_2}{c_4}\right| \;\geqslant\; \frac{L\sqrt{d_{\text{sp}}}}{\gamma} - \epsilon,$$

where $L = 1 + K_2^2$.

Proof. We will first show that the only components of interest are ϕ_1, ϕ_2. Then, we will prove conditions on the amplification of w_{in} over w_{sp} in ϕ_1, ϕ_2. Following is the proof overview:

I. When $w^\star = \mathbf{1}_{d_{\text{in}}}/\sqrt{d_{\text{in}}}$, from the closed form expressions for Σ_A and $\widetilde{\Sigma}$, show that the solution returned by solving the Barlow Twins objective depends on w_{in} and w_{sp} only through the first two components ϕ_1, ϕ_2.

II. For the components ϕ_1, ϕ_2, we will show that the dependence along w_{in} is amplified compared to w_{sp} when the target data sufficiently denoises the spurious feature (*i.e.*, σ_{sp} is sufficiently large).

Part-I:

We can divide the space $\mathbb{R}^d$ into two subspaces that are perpendicular to each other. The first subspace is $\mathcal{W} = \{b_1 \cdot w_{\text{in}} + b_2 \cdot w_{\text{sp}} : b_1, b_2 \in \mathbb{R}\}$, *i.e.* the rank 2 subspace spanned by w_{in} and w_{sp}. The second subspace is $\mathcal{W}_\perp$ where $\mathcal{W}_\perp = \{u \in \mathbb{R}^d : u^\top w_{\text{in}} = 0, u^\top w_{\text{sp}} = 0\}$. Then, from Lemma E.7.4 we can conclude that the matrix Σ_A can be written as:

$$\Sigma_A = \Sigma_{A_{\mathcal{W}}} + \Sigma_{A_{\mathcal{W}_\perp}}$$
$$\Sigma_{A_{\mathcal{W}}} = \frac{1}{4}\begin{bmatrix} \left(\gamma^2(1 + {}^{1}/_{3d_{\text{in}}}) + {}^{\sigma_{\text{in}}^2}/_{3}(1 - {}^{1}/_{d_{\text{in}}})\right) \cdot w^\star {w^\star}^\top, & \gamma\sqrt{{}^{d_{\text{sp}}}/_{2}} \cdot w^\star {w'}^\top \\ \gamma\sqrt{{}^{d_{\text{sp}}}/_{2}} \cdot w' {w^\star}^\top, & \left({}^{d_{\text{sp}}}/_{2} + {}^{4}/_{3} \cdot \sigma_{\text{sp}}^2 + {}^{1}/_{6}\right) \cdot w' {w'}^\top \end{bmatrix}, \quad \text{(E.94)}$$

where $\Sigma_{A_{\mathcal{W}_\perp}} := \mathbb{E}_{a\sim \mathrm{P_A}}\left[\Pi_{\mathcal{W}_\perp}(a)(\Pi_{\mathcal{W}_\perp}(a))^\top\right]$ is the covariance matrix in the null space of $\mathcal{W}$, and $\Pi_{\mathcal{W}_\perp}(a)$ is the projection of augmentation a into the null space of $\mathcal{W}$, *i.e.* the covariance matrix in the space of non-predictive (noise) features. Similarly we can define:

$$\begin{aligned}\widetilde{\Sigma} &= \widetilde{\Sigma}_{\mathcal{W}} + \widetilde{\Sigma}_{\mathcal{W}_\perp} \\ \widetilde{\Sigma}_{\mathcal{W}} &= \frac{1}{4}\begin{bmatrix} \gamma^2 \cdot w^\star {w^\star}^\top, & \gamma\sqrt{d_{\mathrm{sp}}}/2 \cdot w^\star {w'}^\top \\ \gamma\sqrt{d_{\mathrm{sp}}}/2 \cdot w' {w^\star}^\top, & \left(d_{\mathrm{sp}}/2 + \sigma_{\mathrm{sp}}^2/2\right) \cdot w' {w'}^\top \end{bmatrix}\end{aligned} \tag{E.95}$$

Here again $\widetilde{\Sigma}_{\mathcal{W}_\perp} := \mathbb{E}_{x\sim \mathrm{P_U}}\left[\Pi_{\mathcal{W}_\perp}(\mathbb{E}_{c\sim \mathrm{Unif}[0,1]^d}(c \odot x))(\Pi_{\mathcal{W}_\perp}(\mathbb{E}_{c\sim \mathrm{Unif}[0,1]^d}(c \odot x)))^\top\right]$ is the covariance matrix of mean augmentations after they are projected onto the null space of predictive features. The above decomposition also follows from result in Lemma E.7.4.

From Proposition 6.4.3, the closed form expression for the solution returned by optimizing the Barlow Twins objective in (6.3) is $U^\top \Sigma_A^{-1/2}$ where U are the top-k eigenvectors of:

$$\Sigma_A^{-1/2} \cdot \widetilde{\Sigma} \cdot \Sigma_A^{-1/2} \tag{E.96}$$

When $w^\star = \mathbf{1}_{d_{\mathrm{in}}}/\sqrt{d_{\mathrm{in}}}$, then $\Sigma_{A_{\mathcal{W}_\perp}} = \widetilde{\Sigma}_{\mathcal{W}_\perp} + B$ where B is a diagonal matrix with diagonal given by $\frac{1}{3} \cdot \mathrm{diag}(\widetilde{\Sigma}_{\mathcal{W}_\perp})$. Further, since $\mathrm{diag}(\widetilde{\Sigma}_{\mathcal{W}_\perp}) = p \cdot \mathbb{1}_d$ for some constant $p > 0$, the eigenvectors of $\widetilde{\Sigma}_{\mathcal{W}_\perp}$ and $\Sigma_{A_{\mathcal{W}_\perp}}$ are exactly the same. Hence, when we consider the SVD of the expression $\Sigma_A^{-1/2}\widetilde{\Sigma}\Sigma_A^{-1/2}$, the matrices $\Sigma_{A_{\mathcal{W}_\perp}}$ and $\widetilde{\Sigma}_{\mathcal{W}_\perp}$ have no effect on the SVD components that lie along the span of the predictive features. In fact, we only need to consider two rank 2 matrices (first terms in (E.95), (E.94)) and only do the SVD of $\Sigma_{A_{\mathcal{W}}}^{-1/2} \cdot \widetilde{\Sigma}_{\mathcal{W}} \cdot \Sigma_{A_{\mathcal{W}}}^{-1/2}$.

There are only two eigenvectors of $\Sigma_{A_{\mathcal{W}}}^{-1/2} \cdot \widetilde{\Sigma}_{\mathcal{W}} \cdot \Sigma_{A_{\mathcal{W}}}^{-1/2}$. We use λ_1, λ_2 to denote the eigenvalues of $\Sigma_{A_{\mathcal{W}}}$, and $[\cos(\alpha)w^\star, \sin(\alpha)w']^\top$, $[\sin(\alpha)w^\star, -\cos(\alpha)w']^\top$ for the corresponding eigenvectors. Similarly, we use $\widetilde{\lambda}_1, \widetilde{\lambda}_2$ to denote the eigenvalues of $\widetilde{\Sigma}_{\mathcal{W}}$, and $[\cos(\beta)w^\star, \sin(\beta)w']^\top$, $[\sin(\beta)w^\star, -\cos(\beta)w']^\top$ for the corresponding eigenvectors. Let $\mathrm{SVD}_U(\cdot)$ denote the operation of obtaining the singular vectors of a matrix. Then, to compute the components of the final expression: $\mathrm{SVD}_U(\Sigma_A^{-1/2}\widetilde{\Sigma}\Sigma_A^{-1/2})^\top \Sigma_A^{-1/2}$ that lies along the span of predictive features (in $\mathcal{W}$), we need only look at the decomposition of the following matrix:

$$\begin{bmatrix} \cos\theta & , \sin(\theta) \\ \sin\theta & , -\cos(\theta) \end{bmatrix} = \mathrm{SVD}_U\left(\begin{bmatrix} 1/\sqrt{\lambda_1}, & 0 \\ 0, & 1/\sqrt{\lambda_2} \end{bmatrix} \cdot \begin{bmatrix} \cos(\alpha-\beta), & \sin(\alpha-\beta) \\ \sin(\alpha-\beta), & -\cos(\alpha-\beta) \end{bmatrix} \cdot \begin{bmatrix} \sqrt{\widetilde{\lambda}_1}, & 0 \\ 0, & \sqrt{\widetilde{\lambda}_2} \end{bmatrix}\right) \tag{E.97}$$

Based on the above definitions of $\theta, \alpha, \lambda_1, \lambda_2$, we can then formulate ϕ_1 and ϕ_2 in the following way:

$$[\phi_1, \phi_2] = \begin{bmatrix} w^\star \cdot \frac{\cos(\alpha)}{\sqrt{\lambda_1}}, & w^\star \cdot \frac{\sin(\alpha)}{\sqrt{\lambda_2}} \\ w' \cdot \frac{\sin(\alpha)}{\sqrt{\lambda_1}}, & w' \frac{-\cos(\alpha)}{\sqrt{\lambda_2}} \end{bmatrix} \cdot \begin{bmatrix} \cos\theta & , \ \sin(\theta) \\ \sin\theta & , -\cos(\theta) \end{bmatrix} \tag{E.98}$$

To summarize, using arguments in Lemma E.7.4 and the fact that $w^\star = \mathbf{1}_{d_{\text{in}}}/\sqrt{d_{\text{in}}}$, we can afford to focus on just two rank two matrices $\Sigma_{A_\mathcal{W}}, \widetilde{\Sigma}_\mathcal{W}$ in the operation: $\text{SVD}_U(\Sigma_A^{-1/2})\widetilde{\Sigma}\Sigma_A^{-1/2}$. The other singular vectors from the SVD only impact directions that span $\mathcal{W}_\perp$, and the singular vectors obtained by considering only the rank 2 matrices lie only in the space of $\mathcal{W}$.

Part-II:

From the previous part we obtained forms of ϕ_1, ϕ_2 in terms of: $\lambda_1, \lambda_2, \alpha, \theta$, all of which are fully specified by the SVD of $\Sigma_{A_\mathcal{W}}$ and $\widetilde{\Sigma}_\mathcal{W}$. If we define $\tau := \frac{\sqrt{\lambda_1}}{\sqrt{\lambda_2}}$, we can evaluate c_1, c_2, c_3, c_4 as:

$$c_1 = \frac{\cot(\alpha)}{\tau} + \tan(\theta) \tag{E.99}$$

$$c_2 = -1 + \frac{\cot(\alpha)\tan(\theta)}{\tau} \tag{E.100}$$

$$c_3 = \frac{1}{\tau} - \cot(\alpha)\tan(\theta) \tag{E.101}$$

$$c_4 = \frac{\tan(\theta)}{\tau} + \cot(\alpha) \tag{E.102}$$

Now, we are ready to begin proofs for our claims on the amplification factors, *i.e.* on the ratios c_1/c_3, $|c_2/c_4|$.

We will first prove some limiting conditions for c_1/c_3, followed by those on $|c_2/c_4|$. For each of these conditions we will rely on the forms for c_1, c_2, c_3, c_4 derived in the previous part, in terms of α, θ, τ (where $0 \leqslant \alpha, \theta \leqslant \pi/2$). We will also rely on some lemmas that characterize the asymptotic behavior of α, θ and τ as we increase σ_{sp}. We defer the full proof of these helper lemmas to later sections.

Asymptotic behavior of c_1/c_3.

From Lemma E.7.6 and Lemma E.7.7, when $\gamma = {}^{K_1}/_{\sqrt{z}}$ and $\sigma_{\text{sp}} = K_2\sqrt{z}$, then:

$$\lim_{z\to\infty} \frac{c_1}{c_3} = \frac{\cot\alpha + \tau\tan\theta}{1 - \tau\cot\alpha\tan\theta} = \lim_{z\to\infty} \tau\tan\theta = \frac{K_1 K_2^2}{(1 + K_2^2)2\sigma_{\text{in}}^2(1 - {}^1/_{d_{\text{in}}})}, \tag{E.103}$$

where we apply Moore-Osgood when applying limits on intermediate forms. We can do this since $\tau\tan\theta$ approaches a constant, and each of $\cot\alpha, \tau$ and $\tan\theta$ are continuous and smooth functions of z (see Lemma E.7.5).

Asymptotic behavior of $|{}^{c_2}/_{c_4}|$.

When we consider the limiting behavior of ${}^{c_2}/_{c_4 z}$, as we increase z or equivalently σ_{sp} when $\gamma = {}^{K_1}/_{\sqrt{z}}$ and $\sigma_{\text{sp}} = K_2\sqrt{z}$, then we get:

$$\lim_{z\to\infty} \left|\frac{c_2}{c_4 z}\right| = \left|\frac{-1 + \cot(\alpha)\tan(\theta)}{\frac{\tan(\theta)z}{\tau} + \cot(\alpha)z}\right|. \tag{E.104}$$

From Lemma E.7.7, $\cot\alpha\tan\theta \to 0$. Next, if we consider $\lim_{z\to\infty} {}^{z\tan\theta}/_{\tau} = \lim_{z\to\infty} \tau\tan\theta \cdot {}^{z}/_{\tau^2}$. For ${}^{z}/_{\tau^2}$, we invoke Lemma E.7.9, which states that when $\gamma = {}^{K_1}/_{\sqrt{z}}$ and $\sigma_{\text{sp}} = K_2\sqrt{z}$, then:

$$\lim_{z\to\infty} \frac{z}{\tau^2} = \frac{2\sigma_{\text{in}}^2/3(1 - {}^{1}/_{d_{\text{in}}})}{1 + {}^{4}/_{3}K_2^2}. \tag{E.105}$$

Further, in our bound on c_1/c_3, we derived that $\tau\tan\theta \to {}^{K_1K_2^2}/_{(1+K_2^2)2\sigma_{\text{in}}^2(1-{}^{1}/_{d_{\text{in}}})}$. Once again using Moore-Osgood we can plug this along with (E.105) to get:

$$\lim_{z\to\infty} \frac{\tan(\theta)z}{\tau} = \frac{K_1K_2^2}{(1+K_2^2)(3+4K_2^2)}. \tag{E.106}$$

Finally, from Lemma E.7.8, when $\gamma = {}^{K_1}/_{\sqrt{z}}$ and $\sigma_{\text{sp}} = K_2\sqrt{z}$, then:

$$\lim_{z\to\infty} \frac{z}{\tan\alpha} = \frac{K_1}{(1 + {}^{4}/_{3}K_2^2)}. \tag{E.107}$$

Plugging, E.106 and E.107 into E.104 we get the following limit:

$$\lim_{z\to\infty} \left|\frac{c_2}{c_4 z}\right| = \frac{1+K_2^2}{K_1}. \tag{E.108}$$

Since $z = {}^{K_1\sqrt{d_{\text{sp}}}}/_{\gamma}$,

$$\lim_{z\to\infty} \left|\frac{c_2\gamma}{c_4K_1\sqrt{d_{\text{sp}}}}\right| = \frac{1+K_2^2}{K_1} \implies \lim_{z\to\infty} \left|\frac{c_2\gamma}{c_4\sqrt{d_{\text{sp}}}}\right| = 1 + K_2^2 \tag{E.109}$$

Since both ${}^{c_1}/_{c_3}$ and $|{}^{c_2}/_{c_4}|$ are continuous functions of z, with $\liminf_{z\to\infty}$ and $\limsup_{z\to\infty}$ converging to the limits in E.103 and E.104 for both quantities respectively, we conclude that $\forall\epsilon > 0$ there exists σ_{sp_0} such that for all $\sigma_{\text{sp}} \geqslant \sigma_{\text{sp}_0}$, the following is true:

$$\frac{K_1K_2^2d_{\text{in}}}{2L\sigma_{\text{in}}^2(d_{\text{in}}-1)} + \epsilon \geqslant \frac{c_1}{c_3} \geqslant \frac{K_1K_2^2d_{\text{in}}}{2L\sigma_{\text{in}}^2(d_{\text{in}}-1)} - \epsilon \tag{E.110}$$

$$\frac{(1+K_2^2)\sqrt{d_{\text{sp}}}}{\gamma} + \epsilon \geqslant \left|\frac{c_2}{c_4}\right| \geqslant \frac{(1+K_2^2)\sqrt{d_{\text{sp}}}}{\gamma} - \epsilon, \tag{E.111}$$

This completes both Part-I and Part-II of the proof for Theorem 6.4.4.

□

Proof of Corollary 6.4.5

Corollary E.5.8 (CL improves OOD error over ERM but is still imperfect)**.** *For $\gamma, \sigma_{\text{sp}}, d_{\text{sp}}$ defined as in Theorem 6.4.4, $\exists \sigma_{\text{sp}_1}$ such that $\forall \sigma_{\text{sp}} \geqslant \sigma_{\text{sp}_1}$, the target accuracy of CL (linear predictor on Φ_{cl}) is $\geqslant 0.5 \,\text{erfc}\left(-L' \cdot \gamma/\sqrt{2}\sigma_{\text{sp}}\right)$ and $\leqslant 0.5 \,\text{erfc}\left(-4L' \cdot \gamma/\sqrt{2}\sigma_{\text{sp}}\right)$, where $L' = K_2^2 K_1 / \sigma_{\text{in}}^2(1 - 1/d_{\text{in}})$. When $\sigma_{\text{sp}_1} > \sigma_{\text{in}}\sqrt{1 - 1/d_{\text{in}}}$, the lower bound on accuracy is strictly better than ERM from scratch.*

Proof. Recall from Theorem E.5.7, all ϕ_j, for $j \geqslant 3$, lie in the null space of w_{in} and w_{sp}. Since, the predictive features are strictly contained in the rank two space spanned by w_{in} and w_{sp}, without loss of generality we can restrict ourselves to the case where $k = 2$, and when doing training a head $h = [h_1, h_2]^\top \in \mathbb{R}^2$ over contrastive pretrained representations using source labeled data, we get the following max margin solution:

$$\begin{aligned} h_1 &= c_1 \cdot \gamma + c_3 \cdot \sqrt{d_{\text{sp}}} \\ h_2 &= c_2 \cdot \gamma + c_4 \cdot \sqrt{d_{\text{sp}}} \end{aligned} \tag{E.112}$$

Without loss of generality we can divide both h_1 and h_2 by h_1 and get the final classifier to be $\phi_1 + \frac{h_2}{h_1} \cdot \phi_2$:

$$\begin{aligned} &(c_1 w_{\text{in}} + c_3 w_{\text{sp}}) + \frac{h_2}{h_1} \cdot (c_2 w_{\text{in}} + c_4 w_{\text{sp}}) \\ &= (c_1 w_{\text{in}} + c_3 w_{\text{sp}}) + \frac{(c_2\gamma + c_4\sqrt{d_{\text{sp}}})}{(c_1\gamma + c_3\sqrt{d_{\text{sp}}})} \cdot (c_2 w_{\text{in}} + c_4 w_{\text{sp}}) \end{aligned} \tag{E.113}$$

From Lemma E.7.10, we can derive the target accuracy of the classifier h on top of CL representations to be the following:

$$0.5 \,\text{erfc}\left(-\frac{c_1 + \beta c_2}{c_3 + \beta c_4} \cdot \frac{\gamma}{\sqrt{2}\sigma_{\text{sp}}}\right) \tag{E.114}$$

where $\beta = (c_2\gamma + c_4\sqrt{d_{\text{sp}}}) / (c_1\gamma + c_3\sqrt{d_{\text{sp}}})$.

Substituting β into the expression $\frac{c_1 + \beta c_2}{c_3 + \beta c_4}$ we get:

$$\frac{c_1^2\gamma + c_1 c_3\sqrt{d_{\text{sp}}} + c_2^2\gamma + c_2 c_4\sqrt{d_{\text{sp}}}}{c_1 c_3\gamma + c_3^2\sqrt{d_{\text{sp}}} + c_2 c_4\gamma + c_4^2\sqrt{d_{\text{sp}}}} \tag{E.115}$$

We first substitute expressions for c_1, c_2, c_3, c_4 from (E.99), (E.100), (E.101) and (E.102) in the above expression. Then for $\gamma = K_1/\sqrt{z}, \sigma_{\text{sp}} = K_2\sqrt{z}$, we substitute the expressions for $\cot\alpha$, $\tan\theta$, and $\tau = \lambda_1/\lambda_2$ with their corresponding closed form expressions (as functions of z) from Lemma E.7.5. On the resulting expression we apply do repeated applications of L'Hôpital's rule to get the following result:

$$\lim_{z\to\infty} \frac{c_1^2\gamma + c_1c_3\sqrt{d_{\text{sp}}} + c_2^2\gamma + c_2c_4\sqrt{d_{\text{sp}}}}{c_1c_3\gamma + c_3^2\sqrt{d_{\text{sp}}} + c_2c_4\gamma + c_4^2\sqrt{d_{\text{sp}}}} = \frac{2K_2^2K_1}{\sigma_{\text{in}}^2(1 - {}^1\!/_{d_{\text{in}}})} \tag{E.116}$$

Based on $\gamma, d_{\text{sp}}, \sigma_{\text{sp}}$ defined in Theorem 6.4.4, and (E.116) we can conclude that $\exists \sigma_{\text{sp}_1}$ such that for all $\sigma_{\text{sp}} \geqslant \sigma_{\text{sp}_1}$:

$$\frac{4K_2^2K_1}{\sigma_{\text{in}}^2(1 - {}^1\!/_{d_{\text{in}}})} \geqslant \frac{c_1^2\gamma + c_1c_3\sqrt{d_{\text{sp}}} + c_2^2\gamma + c_2c_4\sqrt{d_{\text{sp}}}}{c_1c_3\gamma + c_3^2\sqrt{d_{\text{sp}}} + c_2c_4\gamma + c_4^2\sqrt{d_{\text{sp}}}} \geqslant \frac{K_2^2K_1}{\sigma_{\text{in}}^2(1 - {}^1\!/_{d_{\text{in}}})} \tag{E.117}$$

Finally, applying (E.117) to Lemma E.7.10, we conclude the following: When $\gamma = {}^{K_1K_2}\!/_{\sigma_{\text{sp}}}, d_{\text{sp}} = {}^{\sigma_{\text{sp}}^2}\!/_{K_2^2}$, there exists σ_{sp_1}, such that for any $\sigma_{\text{sp}} \geqslant \sigma_{\text{sp}_1}$, target accuracy of CL is at least $0.5\,\text{erfc}\left(-L' \cdot \frac{\gamma}{\sqrt{2}\sigma_{\text{sp}}}\right)$ and at most $0.5\,\text{erfc}\left(-4L' \cdot \frac{\gamma}{\sqrt{2}\sigma_{\text{sp}}}\right)$, where $L' = \frac{K_2^2K_1}{\sigma_{\text{in}}^2(1-{}^1\!/_{d_{\text{in}}})}$.

Comparison with ERM. Recall from Theorem E.5.1 the performance of ERM classifier (trained from scratch) is $0.5\,\text{erfc}\left(-{}^{\gamma^2}\!/_{\sqrt{2d_{\text{sp}}}\sigma_{\text{sp}}}\right)$. The lower bound on the performance of classifier over CL representations is strictly better than ERM when:

$$\begin{aligned} &\frac{\gamma}{\sqrt{d_{\text{sp}}}} < L' \\ \Longleftarrow\ &\frac{K_2^2K_1}{\sigma_{\text{in}}^2(1 - {}^1\!/_{d_{\text{in}}})} > \frac{\gamma}{\sqrt{d_{\text{sp}}}} \ \Longleftarrow\ \frac{K_2^2K_1}{\sigma_{\text{in}}^2(1 - {}^1\!/_{d_{\text{in}}})} > \frac{K_1K_2^2}{\sigma_{\text{sp}}^2} \\ \Longleftarrow\ &\sigma_{\text{sp}} > \sigma_{\text{in}}\sqrt{1 - {}^1\!/_{d_{\text{in}}}} \ \Longleftarrow\ \sigma_{\text{sp}_1} > \sigma_{\text{in}}\sqrt{1 - {}^1\!/_{d_{\text{in}}}}. \end{aligned}$$

This completes our proof of Corollary 6.4.5.

□

E.5.2 Analysis of STOC: Formal Statement of Theorem 6.4.6

Recall ERM solution over contrastive pretraining. We showed that without loss of generality when k (the output dimensionality of Φ) is greater than 2, we can restrict k to 2 and the Φ can be denoted as $[\phi_1, \phi_2]^\top$ where $\phi_1 = c_1w^\star + c_3w_{\text{sp}}$ and $\phi_2 = c_2w^\star + c_4w_{\text{sp}}$. The ERM solution of the linear head is then given by $h_1, h_2 \in \mathbb{R}$:

$$h_1 = c_1 \cdot \gamma + c_3 \cdot \sqrt{d_{\text{sp}}}\,, \quad \text{and} \quad h_2 = c_2 \cdot \gamma + c_4 \cdot \sqrt{d_{\text{sp}}}\,. \tag{E.118}$$

STOC performs self-training of the linear head over the CL solution. Before introducing the result, we need some additional notation. Let h^t denote the solution of the linear head at iterate t. Without loss of generality, assume that the coefficients in $\phi_1 = c_1w_{\text{in}} + c_3w_{\text{sp}}$ and $\phi_2 = c_2w_{\text{in}} + c_4w_{\text{sp}}$ are such that c_2 is positive and c_1, c_3, and c_4 are negative. Moreover, for simplicity of exposition, assume that $|c_4| > |c_3|$.

Theorem E.5.9. *Under the conditions of Corollary E.5.8 and when* $\frac{\gamma^2}{\sigma_{\mathrm{sp}}} \geqslant \left[\frac{-c_3-c_4}{(c_2+c_1)\cdot|c_1|}\right] \vee \left[\frac{c_4}{c_1\cdot c_2}\right]$*, the target accuracy of ST over CL is lower bounded by* $0.5\cdot\mathrm{erfc}\left(-\left|{}^{c_2}/{}_{c_4}\right|\cdot{}^{\gamma}/{}_{(\sqrt{2}\sigma_{\mathrm{sp}})}\right) \geqslant 0.5\cdot\mathrm{erfc}\left(-L\cdot{}^{\sqrt{d_{\mathrm{sp}}}}/{}_{(\sqrt{2}\sigma_{\mathrm{sp}})}\right)$ *with* $L \geqslant 1$*.*

Before proving Theorem E.5.9, we first connect the condition $\frac{\gamma^2}{\sigma_{\mathrm{sp}}} \geqslant \left[\frac{-c_3-c_4}{(c_2+c_1)\cdot|c_1|}\right] \vee \left[\frac{c_4}{c_1\cdot c_2}\right]$ with the result obtained with contrastive learning.

Remark 1. We first argue that $\left[\frac{-c_3-c_4}{(c_2+c_1)\cdot|c_1|}\right]$ term dominates and hence, if we have $\frac{\gamma^2}{\sigma_{\mathrm{sp}}} \geqslant \left[\frac{-c_3-c_4}{(c_2+c_1)\cdot|c_1|}\right]$, then we get the result in Theorem E.5.9. First, recall that as σ_{sp} increases, we have $\left|\frac{c_3}{c_1}\right|$ converge to $\frac{2L\sigma_{\mathrm{in}}^2(d_{\mathrm{in}}-1)}{K_1K_2^2d_{\mathrm{in}}}$, $c_2 \to 1$ and $\frac{c_1}{c_2} \to 0$. Using these limits, we get:

$$\frac{\gamma^2}{\sigma_{\mathrm{sp}}} = \frac{K_1^2}{K_2\cdot z^{3/2}} \geqslant \frac{2L\sigma_{\mathrm{in}}^2(d_{\mathrm{in}}-1)}{K_1K_2^2d_{\mathrm{in}}}\,. \tag{E.119}$$

which reduces the following condition: $d_{\mathrm{sp}} \leqslant K_1^2K_2^{2/3}\cdot\left(\frac{d_{\mathrm{in}}}{2L\sigma_{\mathrm{in}}^2(d_{\mathrm{in}}-1)}\right)^{2/3}$.

Proof. First, we create an outline of the proof. We argue about the updates of h^t showing that both h_1^t and h_2^t increase with $|h_2^t|$ becoming greater than $|h_1^t|$ for some large t. Then we show that $|h_2^t| \geqslant |h_1^t|$ is sufficient to obtain near-perfect target generalization.

Part 1. Recall the loss of used for self-training of h:

$$\begin{aligned}\mathcal{L}_{\mathrm{st}}(h) &= \mathbb{E}_{\mathrm{P}_t(x)}\left[\ell(h^\top\Phi x, \mathrm{sgn}(h^\top\Phi x))\right] && \text{(E.120)}\\ &= \mathbb{E}_{\mathrm{P}_t(x)}\left[\exp\left(-\left|h^\top\Phi x\right|\right)\right] && \text{(E.121)}\\ &= \mathbb{E}_{z\sim\mathcal{N}(0,1)}\left[\exp\left(-\left|c_1\gamma h_1 + c_2\gamma h_2 + (c_3\sigma_{\mathrm{sp}}h_1 + c_4\sigma_{\mathrm{sp}}h_2)\cdot z\right|\right)\right]. && \text{(E.122)}\end{aligned}$$

Define $\mu_t = c_1\gamma h_1^t + c_2\gamma h_2^t$ and $\sigma_t = c_3\sigma_{\mathrm{sp}}h_1^t + c_4\sigma_{\mathrm{sp}}h_2^t$. With this notation, we can re-write the loss in (E.122) as $\mathcal{L}_{\mathrm{st}}(h^t) = \mathbb{E}_{z\sim\mathcal{N}(0,\sigma_t^2)}\left[\exp\left(-\left|\mu_t + z\right|\right)\right]$.

Similar to the the treatment in Theorem E.5.2, we now derive a closed-form expression of $\mathcal{L}_{\mathrm{st}}(h^t)$ in Lemma E.7.11:

$$\mathcal{L}_{\mathrm{st}}(h^t) = \frac{1}{2}\left(\exp\left(\frac{\sigma_t^2}{2}-\mu_t\right)\cdot\mathrm{erfc}\left(-\frac{\mu_t}{\sqrt{2}\sigma_t}+\frac{\sigma_t}{\sqrt{2}}\right) + \exp\left(\frac{\sigma_t^2}{2}+\mu_t\right)\cdot\mathrm{erfc}\left(\frac{\mu_t}{\sqrt{2}\sigma_t}+\frac{\sigma_t}{\sqrt{2}}\right)\right). \tag{E.123}$$

Define:

$$A_1(\mu_t,\sigma_t) = \exp\left(\frac{\sigma_t^2}{2}-\mu_t\right)\cdot\mathrm{erfc}\left(-\frac{\mu_t}{\sqrt{2}\sigma_t}+\frac{\sigma_t}{\sqrt{2}}\right)$$

$$= \sqrt{\frac{2}{\pi}} \exp\left(-\frac{\mu_t^2}{2\sigma_t^2}\right) \mathrm{r}\left(\sigma_t - \frac{\mu_t}{\sigma_t}\right) \,, \tag{E.124}$$

$$A_2(\mu_t, \sigma_t) = \exp\left(\frac{\sigma_t^2}{2} + \mu_t\right) \cdot \mathrm{erfc}\left(\frac{\mu_t}{\sqrt{2}\sigma_t} + \frac{\sigma_t}{\sqrt{2}}\right)$$

$$= \sqrt{\frac{2}{\pi}} \exp\left(-\frac{\mu_t^2}{2\sigma_t^2}\right) \mathrm{r}\left(\sigma_t + \frac{\mu_t}{\sigma_t}\right) \,, \tag{E.125}$$

$$A_3(\mu_t, \sigma_t) = \frac{2\sqrt{2}}{\sqrt{\pi}} \exp\left(-\frac{\mu_t^2}{2\sigma_t^2}\right) \,. \tag{E.126}$$

Let $\widetilde{h}^{t+1}$ denote the un-normalized gradient descent update at iterate $t+1$. We have:

$$\widetilde{h}^{t+1} = h^t - \eta \cdot \frac{\partial \mathcal{L}_{\mathrm{st}}(h^t)}{\partial h} \,. \tag{E.127}$$

Now we will individually argue about the update of $\widetilde{h}^{t+1}$. First, we have:

$$\widetilde{h}_1^{t+1} = h_1^t - \eta \cdot \frac{\partial \mathcal{L}_{\mathrm{st}}(h^t)}{\partial h_1}$$

$$\widetilde{h}_1^{t+1} = h_1^t - \eta \cdot \underbrace{\left[A_1 \cdot (\sigma_t c_3 \sigma_{\mathrm{sp}} - c_1\gamma) + A_2 \cdot (\sigma_t c_3 \sigma_{\mathrm{sp}} + c_1\gamma) - A_3 c_3 \sigma_{\mathrm{sp}}\right]}_{\delta_1} \,. \tag{E.128}$$

and second, we have:

$$\widetilde{h}_2^{t+1} = h_2^t - \eta \cdot \frac{\partial \mathcal{L}_{\mathrm{st}}(h^t)}{\partial h_2}$$

$$\widetilde{h}_2^{t+1} = h_2^t - \eta \cdot \underbrace{\left[A_1 \cdot (\sigma_t c_4 \sigma_{\mathrm{sp}} - c_2\gamma) + A_2 \cdot (\sigma_t c_4 \sigma_{\mathrm{sp}} + c_2\gamma) - A_3 c_4 \sigma_{\mathrm{sp}}\right]}_{\delta_2} \,. \tag{E.129}$$

We will now argue the conditions under which h_2^{t+1} increases till its value reaches $1/\sqrt{2}$. In particular, we will argue that when h_2^t is negative, the norm $|h_2^t|$ decreases and when h_2^t becomes positive, then its norm increases. We show that the following three conditions are sufficient to argue the increasing value of h_2^t: for all t, we have (i) $\mu_t \geqslant \mu_c$ and $|\sigma_t| < \sigma_c$ for constant $\mu_c = |c_1 \cdot \gamma|/2$ and $\sigma_c = |c_4\sigma_{\mathrm{sp}}|$; (ii) $\delta_2 < 0$; (iii) $|\delta_2| \geqslant \delta_1$. In Lemma E.5.11, we argue that our assumption on the initialization of the backbone learned with BT implies the previous three conditions.

Case-1. When h_2^t is negative (and after the update, it remains negative). Then we want to argue the following:

$$\frac{(h_2^t - \eta\delta_2)^2}{(h_2^t - \eta\delta_2)^2 + (h_1^t - \eta\delta_1)^2} \leqslant (h_2^t)^2 \tag{E.130}$$

$$\Rightarrow \quad \frac{(h_2^t - \eta\delta_2)^2}{(h_2^t)^2} \leqslant (h_2^t - \eta\delta_2)^2 + (h_1^t - \eta\delta_1)^2 \tag{E.131}$$

$$\Rightarrow \quad \frac{{h_2^t}^2 + \eta^2\delta_2^2 - 2\eta\delta_2 h_2^t}{(h_2^t)^2} \leqslant {h_2^t}^2 + \eta^2\delta_2^2 - 2\eta h_2^t\delta_2 + {h_1^t}^2 + \eta^2\delta_1^2 - 2\eta h_1^t\delta_1 \tag{E.132}$$

$$\Rightarrow \quad 1 + \frac{\eta^2\delta_2^2 - 2\eta\delta_2 h_2^t}{(h_2^t)^2} \leqslant 1 + \eta^2\delta_2^2 - 2\eta h_2^t\delta_2 + \eta^2\delta_1^2 - 2\eta h_1^t\delta_1 \tag{E.133}$$

$$\Rightarrow \quad \eta^2\delta_2^2 - 2\eta\delta_2 h_2^t \leqslant \left[\eta^2\delta_2^2 - 2\eta h_2^t\delta_2 + \eta^2\delta_1^2 - 2\eta h_1^t\delta_1\right](h_2^t)^2 \tag{E.134}$$

$$\Rightarrow \quad \eta^2\delta_2^2(h_1^t)^2 - 2\eta\delta_2 h_2^t(h_1^t)^2 \leqslant \eta^2\delta_1^2(h_2^t)^2 - 2\eta h_1^t\delta_1(h_2^t)^2 \tag{E.135}$$

$$\Rightarrow \quad \eta^2\delta_2^2(h_1^t)^2 - \eta^2\delta_1^2(h_2^t)^2 \leqslant 2\eta\delta_2 h_2^t(h_1^t)^2 - 2\eta h_1^t\delta_1(h_2^t)^2 \tag{E.136}$$

$$\Rightarrow \quad \left[\eta\delta_2(h_1^t) - \eta\delta_1(h_2^t)\right]\left[\eta\delta_2(h_1^t) + \eta\delta_1(h_2^t)\right] \leqslant 2h_2^t h_1^t\left[\eta\delta_2(h_1^t) - \eta\delta_1(h_2^t)\right] \tag{E.137}$$

$$\Rightarrow \quad \left[\eta\delta_2(h_1^t) + \eta\delta_1(h_2^t)\right] \leqslant 2h_2^t h_1^t \tag{E.138}$$

Since $\delta_2 < 0$, $|\delta_2| \geqslant |\delta_1|$ and $h_2^t < h_1^t < 0$, we have $[\eta\delta_2(h_1^t) - \eta\delta_1(h_2^t)]$ as positive. This implies inequality (E.137) to (E.138) and for small enough η, (E.138) will continue to hold true.

Case-2. When h_2^t is positive but less than $1/\sqrt{2}$. Then we want to argue the following:

$$\frac{(h_2^t - \eta\delta_2)^2}{(h_2^t - \eta\delta_2)^2 + (h_1^t - \eta\delta_1)^2} \geqslant (h_2^t)^2 \tag{E.139}$$

$$\Rightarrow \quad \frac{(h_2^t - \eta\delta_2)^2}{(h_2^t)^2} \geqslant (h_2^t - \eta\delta_2)^2 + (h_1^t - \eta\delta_1)^2 \tag{E.140}$$

$$\Rightarrow \quad \frac{{h_2^t}^2 + \eta^2\delta_2^2 - 2\eta\delta_2 h_2^t}{(h_2^t)^2} \geqslant {h_2^t}^2 + \eta^2\delta_2^2 - 2\eta h_2^t\delta_2 + {h_1^t}^2 + \eta^2\delta_1^2 - 2\eta h_1^t\delta_1 \tag{E.141}$$

$$\Rightarrow \quad 1 + \frac{\eta^2\delta_2^2 - 2\eta\delta_2 h_2^t}{(h_2^t)^2} \geqslant 1 + \eta^2\delta_2^2 - 2\eta h_2^t\delta_2 + \eta^2\delta_1^2 - 2\eta h_1^t\delta_1 \tag{E.142}$$

$$\Rightarrow \quad \eta^2\delta_2^2 - 2\eta\delta_2 h_2^t \geqslant \left[\eta^2\delta_2^2 - 2\eta h_2^t\delta_2 + \eta^2\delta_1^2 - 2\eta h_1^t\delta_1\right](h_2^t)^2 \tag{E.143}$$

$$\Rightarrow \quad \eta^2\delta_2^2(h_1^t)^2 - 2\eta\delta_2 h_2^t(h_1^t)^2 \geqslant \eta^2\delta_1^2(h_2^t)^2 - 2\eta h_1^t\delta_1(h_2^t)^2 \tag{E.144}$$

$$\Rightarrow \quad \eta^2\delta_2^2(h_1^t)^2 - \eta^2\delta_1^2(h_2^t)^2 \geqslant 2\eta\delta_2 h_2^t(h_1^t)^2 - 2\eta h_1^t\delta_1(h_2^t)^2 \tag{E.145}$$

$$\Rightarrow \quad \left[\eta\delta_2(h_1^t) - \eta\delta_1(h_2^t)\right]\left[\eta\delta_2(h_1^t) + \eta\delta_1(h_2^t)\right] \geqslant 2h_2^t h_1^t\left[\eta\delta_2(h_1^t) - \eta\delta_1(h_2^t)\right] \tag{E.146}$$

$$\Rightarrow \quad \left[\eta\delta_2(h_1^t) + \eta\delta_1(h_2^t)\right] \geqslant 2h_2^t h_1^t \tag{E.147}$$

Since $\delta_2 < 0$, $|\delta_2| \geqslant |\delta_1|$, $h_1^t \leqslant -1/\sqrt{2}$ and $0 < h_2^t < 1/\sqrt{2}$, we have $[\eta\delta_2(h_1^t) - \eta\delta_1(h_2^t)]$ as positive. This implies inequality (E.146) to (E.147). Focusing on (E.147), we note that

$h_1^t \cdot \delta_2$ is positive and greater in magnitude than $h_2^t \cdot \delta_1$. Moreover, since $h_2^t h_1^t$ is negative, (E.147) will continue to hold true.

Now, when h_2^t is positive and greater than $1/\sqrt{2}$, then h_2^t will stay in that region. Convergence of STOC together with conditions of convergence as in Lemma E.5.10 will imply that the at convergence h_2^t will remain greater than $1/\sqrt{2}$, such that $\frac{h_1^{t_c}}{h_2^{t_c}} = \frac{\delta_1}{\delta_2}$. Now we bound the target error of STOC.

Part 2. To bound the accuracy at any iterate t when $h_2^t \geqslant 1/\sqrt{2}$, we have from Lemma E.7.10:

$$\mathbb{E}_{\mathrm{P}_t}\left[y \cdot \left({h^t}^\top \phi_{\mathrm{cl}} x\right) > 0\right] = \mathbb{E}_{z \sim \mathcal{N}(0,1)}\left[z > -\frac{c_1 \gamma h_1^t + c_2 \gamma h_2^t}{|c_3 \sigma_{\mathrm{sp}} h_1^t + c_4 \sigma_{\mathrm{sp}} h_2^t|}\right]. \tag{E.148}$$

We now upper bound and lower bound the fraction $\frac{c_1 \gamma h_1^t + c_2 \gamma h_2^t}{|c_3 \sigma_{\mathrm{sp}} h_1^t + c_4 \sigma_{\mathrm{sp}} h_2^t|}$ in RHS in (E.148): (i) $c_1 \gamma h_1^t + c_2 \gamma h_2^t \geqslant c_2 \gamma h_2^t$ since both $c_1 \gamma h_1^t$ and $c_2 \gamma h_2^t$ have same sign; (ii) $|c_3 \sigma_{\mathrm{sp}} h_1^t + c_4 \sigma_{\mathrm{sp}} h_2^t| \leqslant |c_4 \sigma_{\mathrm{sp}} h_2^t|$ because $|c_4 \sigma_{\mathrm{sp}} h_2^t| \geqslant |c_3 \sigma_{\mathrm{sp}} h_1^t|$ and they have opposite signs. Hence, from (E.148), we have:

$$\mathbb{E}_{\mathrm{P}_t}\left[y \cdot \left({h^t}^\top \phi_{\mathrm{cl}} x\right) > 0\right] = \mathbb{E}_{z \sim \mathcal{N}(0,1)}\left[z > -\frac{c_2 \gamma h_2^t}{|c_4 \sigma_{\mathrm{sp}} h_2^t|}\right] = \mathbb{E}_{z \sim \mathcal{N}(0,1)}\left[z > -\frac{c_2 \gamma}{|c_4 \sigma_{\mathrm{sp}}|}\right]. \tag{E.149}$$

Substituting the definition of erfc, the expression (E.149) gives us the required lower bound on the target accuracy.

□

Lemma E.5.10 (Convergence of STOC)**.** *Assume the gradient updates as in* (E.128) *and* (E.129)*. Then STOC converges at $t = t_c$ when $\frac{h_1^{t_c}}{h_2^{t_c}} = \frac{\delta_1}{\delta_2}$. For $t > t_c$,* (E.128) *and* (E.129) *make no updates to the linear h.*

Proof. When the gradient updates δ_1 and δ_2 are such that h_1^{t+1} matches h_1^t, we have convergence of STOC.

$$\frac{(h_2^t - \eta\delta_2)^2}{(h_2^t - \eta\delta_2)^2 + (h_1^t - \eta\delta_1)^2} = (h_2^t)^2 \tag{E.150}$$

$$\Rightarrow \quad \frac{(h_2^t - \eta\delta_2)^2}{(h_2^t)^2} = (h_2^t - \eta\delta_2)^2 + (h_1^t - \eta\delta_1)^2 \tag{E.151}$$

$$\Rightarrow \quad \frac{{h_2^t}^2 + \eta^2\delta_2^2 - 2\eta\delta_2 h_2^t}{(h_2^t)^2} = {h_2^t}^2 + \eta^2\delta_2^2 - 2\eta h_2^t \delta_2 + {h_1^t}^2 + \eta^2\delta_1^2 - 2\eta h_1^t \delta_1 \tag{E.152}$$

$$\Rightarrow \qquad 1+\frac{\eta^2\delta_2^2 - 2\eta\delta_2 h_2^t}{(h_2^t)^2} = 1+\eta^2\delta_2^2 - 2\eta h_2^t\delta_2 + \eta^2\delta_1^2 - 2\eta h_1^t\delta_1 \tag{E.153}$$

$$\Rightarrow \qquad \eta^2\delta_2^2 - 2\eta\delta_2 h_2^t = \left[\eta^2\delta_2^2 - 2\eta h_2^t\delta_2 + \eta^2\delta_1^2 - 2\eta h_1^t\delta_1\right](h_2^t)^2 \tag{E.154}$$

$$\Rightarrow \qquad \eta^2\delta_2^2(h_1^t)^2 - 2\eta\delta_2 h_2^t(h_1^t)^2 = \eta^2\delta_1^2(h_2^t)^2 - 2\eta h_1^t\delta_1(h_2^t)^2 \tag{E.155}$$

$$\Rightarrow \qquad \eta^2\delta_2^2(h_1^t)^2 - \eta^2\delta_1^2(h_2^t)^2 = 2\eta\delta_2 h_2^t(h_1^t)^2 - 2\eta h_1^t\delta_1(h_2^t)^2 \tag{E.156}$$

$$\Rightarrow \qquad \left[\eta\delta_2(h_1^t) - \eta\delta_1(h_2^t)\right]\left[\eta\delta_2(h_1^t) + \eta\delta_1(h_2^t)\right] = 2h_2^t h_1^t\left[\eta\delta_2(h_1^t) - \eta\delta_1(h_2^t)\right] \tag{E.157}$$

Thus either $[\eta\delta_2(h_1^t) - \eta\delta_1(h_2^t)] = 0$ or $[\eta\delta_2(h_1^t) + \eta\delta_1(h_2^t)] = 2h_2^t h_1^t$. Since η is such that $h_1 - \eta\delta_1 < 0$, $[\eta\delta_2(h_1^t) + \eta\delta_1(h_2^t)] \neq 2h_2^t h_1^t$ implying that $[\eta\delta_2(h_1^t) - \eta\delta_1(h_2^t)] = 0$ giving us the required condition. □

Lemma E.5.11. *Under the initialization conditions assumed in Theorem E.5.9, for all t, we have: (i) $\mu_t \geqslant \mu_c$ and $|\sigma_t| \leqslant \sigma_c$ for constant $\mu_c = |c_1 \cdot \gamma|/2$ and $\sigma_c = |c_4\sigma_{\text{sp}}|$; (ii) $\delta_2 < 0$; (iii) $|\delta_2| \geqslant \delta_1$, where $\delta_1 = A_1 \cdot (\sigma_t c_3 \sigma_{\text{sp}} - c_1\gamma) + A_2 \cdot (\sigma_t c_3\sigma_{\text{sp}} + c_1\gamma) - A_3 c_3 \sigma_{\text{sp}}$ and $\delta_2 = A_1 \cdot (\sigma_t c_4 \sigma_{\text{sp}} - c_2\gamma) + A_2 \cdot (\sigma_t c_4 \sigma_{\text{sp}} + c_2\gamma) - A_3 c_4 \sigma_{\text{sp}}$ for A_1, A_2 and A_3 defined in (E.124), (E.125), and (E.126).*

Proof. Recall, $\mu_t = c_1\gamma h_1^t + c_2\gamma h_2^t$ and $\sigma_t = c_3\sigma_{\text{sp}} h_1^t + c_4\sigma_{\text{sp}} h_2^t$. First, we argue that μ_t increases from the initialization value. Notice that $\mu_0 = c_1\gamma h_1^0 + c_2\gamma h_2^0$. Due to Corollary E.5.8, we have $h_2^0 \geqslant 0$. And since $|c_2| > |c_1|$, we get $\mu_0 \geqslant |c_1\gamma|$ as both c_1 and h_1^0 are of same sign. Moreover, as training progresses with h_1^t remaining negative and h_2^t remaining positive, we have μ_t stays greater than μ_0.

Recall the definition of A_1, A_2, and A_3 in (E.124), (E.125), and (E.126). Moreover, recall the definition of $\alpha_1(\mu_t, \sigma_t)$ and $\alpha_2(\mu_t, \sigma_t)$:

$$\alpha_1(\mu_t, \sigma_t) = \sqrt{\frac{2}{\pi}}\exp\left(-\frac{\mu_t^2}{2\sigma_t^2}\right)\left[\mathrm{r}\left(\sigma_t + \frac{\mu_t}{\sigma_t}\right) - \mathrm{r}\left(\sigma_t - \frac{\mu_t}{\sigma_t}\right)\right]. \tag{E.158}$$

and

$$\alpha_2(\mu_t, \sigma_t) = \sqrt{\frac{2}{\pi}}\exp\left(-\frac{\mu_t^2}{2\sigma_t^2}\right)\left[\mathrm{r}\left(\sigma_t + \frac{\mu_t}{\sigma_t}\right) + \mathrm{r}\left(\sigma_t - \frac{\mu_t}{\sigma_t}\right) - \frac{2}{\sigma_t}\right]. \tag{E.159}$$

Thus, we have $\alpha_1(\mu_t, \sigma_t) \cdot A_3 = A_1 \cdot \sigma_t$ and $\alpha_2(\mu_t, \sigma_t) \cdot A_3 = \sigma_t \cdot \left(A_2 \cdot -\frac{2}{\sigma_t} A_3\right)$. Replacing the definition of A_1, A_2, and A_3 in δ_1 and δ_2, we get:

$$\delta_1 = \sigma_t c_3 \sigma_{\text{sp}} \cdot \alpha_2(\mu_t, \sigma_t) + c_1\gamma\alpha_1(\mu_t, \sigma_t) \quad \text{and} \quad \delta_2 = \sigma_t c_4\sigma_{\text{sp}} \cdot \alpha_2(\mu_t, \sigma_t) + c_2\gamma\alpha_1(\mu_t, \sigma_t) \tag{E.160}$$

We now upper bound and lower bound α_1 and α_2 by using the properties of $\mathrm{r}(\cdot)$. We use Taylor's expansion on $\mathrm{r}(\cdot)$ and we get:

$$\mathrm{r}(\sigma_t) + \mathrm{r}'(\sigma_t) \cdot \left(\frac{\mu_t}{\sigma_t}\right) \leqslant \mathrm{r}\left(\sigma_t + \frac{\mu_t}{\sigma_t}\right) \leqslant \mathrm{r}(\sigma_t) + \mathrm{r}'(\sigma_t) \cdot \left(\frac{\mu_t}{\sigma_t}\right) + \mathrm{r}''(\sigma_t) \cdot \left(\frac{\mu_t}{\sigma_t}\right)^2 \tag{E.161}$$

and similarly, we get:

$$\mathrm{r}(\sigma_t) - \mathrm{r}'(\sigma_t) \cdot \left(\frac{\mu_t}{\sigma_t}\right) + \mathrm{r}''(\sigma_t) \cdot \left(\frac{\mu_t}{\sigma_t}\right)^2 \leqslant \mathrm{r}\left(\sigma_t - \frac{\mu_t}{\sigma_t}\right) \leqslant \mathrm{r}(\sigma_t) - \mathrm{r}'(\sigma_t) \cdot \left(\frac{\mu_t}{\sigma_t}\right) + R''\left(\frac{\mu_t}{\sigma_t}\right)^2 \tag{E.162}$$

where $R'' = \mathrm{r}''(\sigma_0)$. This is because $\mathrm{r}''(\cdot)$ takes positive values and is a decreasing function in σ_t (refer to Lemma E.7.2). We now lower bound $\alpha_1(\mu_t, \sigma_t)$ and upper bound $\alpha_2(\mu_t, \sigma_t)$:

$$\frac{\alpha_1(\mu_t, \sigma_t)}{\sqrt{\frac{2}{\pi}\exp\left(-\frac{\mu_t^2}{2\sigma_t^2}\right)}} \leqslant 2\mathrm{r}'(\sigma_t) \cdot \left(\frac{\mu_t}{\sigma_t}\right) \tag{E.163}$$

$$\frac{\alpha_2(\mu_t, \sigma_t)}{\sqrt{\frac{2}{\pi}\exp\left(-\frac{\mu_t^2}{2\sigma_t^2}\right)}} \geqslant 2\mathrm{r}(\sigma_t) + \mathrm{r}''(\sigma_t) \cdot \left(\frac{\mu_t}{\sigma_t}\right)^2 - \frac{2}{\sigma_t} \tag{E.164}$$

Part-1. We first prove that $\delta_2 \leqslant 0$. Substituting the lower bound and upper bound in (E.160) gives us the following as stricter a sufficient condition (i.e., (E.165) implies $\delta_2 \leqslant 0$):

$$\left[2\mathrm{r}(\sigma_t) + \mathrm{r}''(\sigma_t) \cdot \left(\frac{\mu_t}{\sigma_t}\right)^2 - \frac{2}{\sigma_t}\right] \cdot \frac{\sigma_{\mathrm{sp}} \cdot (-c_4)}{\gamma \cdot c_2} \geqslant 2\mathrm{r}'(\sigma_t) \cdot \left(\frac{\mu_t}{\sigma_t}\right) \tag{E.165}$$

$$\iff \left[2\mathrm{r}(\sigma_t) + \mathrm{r}''(\sigma_t) \cdot \left(\frac{\mu_t}{\sigma_t}\right)^2 - \frac{2}{\sigma_t}\right] \geqslant 2\mathrm{r}'(\sigma_t) \cdot \left(\frac{\mu_t}{\sigma_t}\right) \cdot \frac{\gamma \cdot c_2}{\sigma_{\mathrm{sp}} \cdot (-c_4)} \tag{E.166}$$

$$\iff 2\mathrm{r}(\sigma_t) + \mathrm{r}''(\sigma_t) \cdot \left(\frac{\mu_t}{\sigma_t}\right)^2 - \frac{2}{\sigma_t} - 2\mathrm{r}'(\sigma_t) \cdot \left(\frac{\mu_t}{\sigma_t}\right) \cdot \frac{\gamma \cdot c_2}{\sigma_{\mathrm{sp}} \cdot (-c_4)} \geqslant 0 \tag{E.167}$$

$$\iff 2\mathrm{r}(\sigma_t) \cdot \sigma_t + \mathrm{r}''(\sigma_t) \cdot \frac{\mu_t^2}{\sigma_t} - 2 - 2\mathrm{r}'(\sigma_t) \cdot \mu_t \cdot \frac{\gamma \cdot c_2}{\sigma_{\mathrm{sp}} \cdot (-c_4)} \geqslant 0 \tag{E.168}$$

$$\iff 2\mathrm{r}'(\sigma_t) + \mathrm{r}''(\sigma_t) \cdot \frac{\mu_t^2}{\sigma_t} - 2\mathrm{r}'(\sigma_t) \cdot \mu_t \cdot \frac{\gamma \cdot c_2}{\sigma_{\mathrm{sp}} \cdot (-c_4)} \geqslant 0 \tag{E.169}$$

$$\iff \mathrm{r}''(\sigma_t) \cdot \frac{\mu_t^2}{\sigma_t} + 2\mathrm{r}'(\sigma_t) \cdot \left[1 - \mu_t \cdot \frac{\gamma \cdot c_2}{\sigma_{\mathrm{sp}} \cdot (-c_4)}\right] \geqslant 0 \tag{E.170}$$

Thus, if we have $\mu_t \geqslant \frac{\sigma_{\mathrm{sp}} \cdot (-c_4)}{\gamma \cdot c_2}$, then (E.165) holds true.

Part-2. Next, we prove that $|\delta_2| \geqslant \delta_1$. Substituting the lower bound and upper bound in (E.160) gives us the following as stricter a sufficient condition (i.e., (E.171) implies $|\delta_2| \geqslant \delta_1$):

$$\left[2\mathrm{r}\left(\sigma_t\right) + \mathrm{r}''\left(\sigma_t\right) \cdot \left(\frac{\mu_t}{\sigma_t}\right)^2 - \frac{2}{\sigma_t}\right] \cdot \frac{\sigma_{\mathrm{sp}} \cdot (-c_4 - c_3)}{\gamma \cdot (c_2 + c_1)} \geqslant 2\mathrm{r}'\left(\sigma_t\right) \cdot \left(\frac{\mu_t}{\sigma_t}\right) \tag{E.171}$$

$$\Longleftrightarrow \left[2\mathrm{r}\left(\sigma_t\right) + \mathrm{r}''\left(\sigma_t\right) \cdot \left(\frac{\mu_t}{\sigma_t}\right)^2 - \frac{2}{\sigma_t}\right] \geqslant 2\mathrm{r}'\left(\sigma_t\right) \cdot \left(\frac{\mu_t}{\sigma_t}\right) \cdot \frac{\gamma \cdot (c_2 + c_1)}{\sigma_{\mathrm{sp}} \cdot (-c_4 - c_3)} \tag{E.172}$$

$$\Longleftrightarrow 2\mathrm{r}\left(\sigma_t\right) + \mathrm{r}''\left(\sigma_t\right) \cdot \left(\frac{\mu_t}{\sigma_t}\right)^2 - \frac{2}{\sigma_t} - 2\mathrm{r}'\left(\sigma_t\right) \cdot \left(\frac{\mu_t}{\sigma_t}\right) \cdot \frac{\gamma \cdot (c_2 + c_1)}{\sigma_{\mathrm{sp}} \cdot (-c_4 - c_3)} \geqslant 0 \tag{E.173}$$

$$\Longleftrightarrow 2\mathrm{r}\left(\sigma_t\right) \cdot \sigma_t + \mathrm{r}''\left(\sigma_t\right) \cdot \frac{\mu_t^2}{\sigma_t} - 2 - 2\mathrm{r}'\left(\sigma_t\right) \cdot \mu_t \cdot \frac{\gamma \cdot (c_2 + c_1)}{\sigma_{\mathrm{sp}} \cdot (-c_4 - c_3)} \geqslant 0 \tag{E.174}$$

$$\Longleftrightarrow 2\mathrm{r}'\left(\sigma_t\right) + \mathrm{r}''\left(\sigma_t\right) \cdot \frac{\mu_t^2}{\sigma_t} - 2\mathrm{r}'\left(\sigma_t\right) \cdot \mu_t \cdot \frac{\gamma \cdot (c_2 + c_1)}{\sigma_{\mathrm{sp}} \cdot (-c_4 - c_3)} \geqslant 0 \tag{E.175}$$

$$\Longleftrightarrow \mathrm{r}''\left(\sigma_t\right) \cdot \frac{\mu_t^2}{\sigma_t} + 2\mathrm{r}'\left(\sigma_t\right) \cdot \left[1 - \mu_t \cdot \frac{\gamma \cdot (c_2 + c_1)}{\sigma_{\mathrm{sp}} \cdot (-c_4 - c_3)}\right] \geqslant 0 \tag{E.176}$$

Thus, if we have $\mu_t \geqslant \frac{\sigma_{\mathrm{sp}} \cdot (-c_4 - c_3)}{\gamma \cdot (c_2 + c_1)}$, then (E.171) holds true which in-turn implies $|\delta_2| \geqslant \delta_1$. Plugging in $\mu_t \geqslant \mu_0$, we get the required condition.

□

E.5.3 Analysis for SSL

For SSL analysis, we argue that the projection learned by contrastive pretraining can significantly improve the generalization of the linear head learned on top, leaving little to no room for improvement for self-training. Our analysis leverages the margin-based bound for linear models from Kakade et al. (2008). Before introducing the result, we present some additional notation. Let $\mathrm{Err}_D(w)$ denote 0-1 error of a classifier on a distribution D. Define 0-1 error with margin ξ as $\widehat{w}\mathrm{Err}_\xi(w) = \sum_{i=1}^n \frac{\mathbb{I}\left[y_i w^\top x_i \leqslant \xi\right]}{n}$.

Theorem E.5.12 (generalization bound for margin loss)**.** *For all classifiers w and margin γ, we have with probability at least $1 - \delta$:*

$$\mathrm{Err}_T(w) \leqslant \widehat{w}\mathrm{Err}_\xi(w) + 4\frac{B}{\xi}\sqrt{\frac{1}{n}} + \sqrt{\frac{\log(2/\delta)}{n}} + \sqrt{\frac{\log(\log_2(4B/\xi))}{n}}, \tag{E.177}$$

where $B = 4\max(\max(\sigma_{\mathrm{in}}, \sigma_{\mathrm{sp}}), 1) \cdot \left(\sqrt{d_{\mathrm{in}} + d_{\mathrm{sp}}} + \sqrt{\log\left({}^{2n}/_{\delta}\right)}\right) + \gamma$ is a high probability upper bound on the ℓ_2 norm of the input points x.

Proof. The result is a trivial application of union bound over: (1) Corollary 6 in Kakade et al. (2008); and (2) high probability bound over norms of sub-gaussian random variables (Sec. 5.2 in (Wainwright, 2019)). □

When $\widehat{w}\text{Err}_\xi(w)$ is close to zero, the denominating term in RHS of (E.177) is ${}^{4B}\!/\!{}_{\xi}\sqrt{{}^{1}\!/\!{}_{n}}$. From Proposition 6.4.3, CL solution ϕ_{cl} obtained on the target domain alone (for SSL setup) is w_{in} when $k = 1$. Intuitively, since the target data has only one predictive feature (along w_{in}), CL directly recovers this predictive feature as it is the predominant direction that minimizes invariance loss. Consequently, projecting the inputs on the CL solution mainly reduces the value of B on the projected data. This happens because the effective dimension is reduced from $\sqrt{d} = \sqrt{d_{\text{in}} + d_{\text{sp}}}$ to $\sqrt{k}$ (which is $= 1$ for $k = 1$), which is the output dimension of the feature extractor ϕ_{cl}. Additionally, since w_{in} is recovered by ϕ_{cl}, the maximum margin between the two classes remains γ, thus for any $\xi \leqslant \gamma$, $\exists w$ such that $\widehat{w}\text{Err}_\xi(w) = 0$.

Assuming we can recover the linear predictor that minimizes the empirical loss, the only dominating term left in the upper bound in (E.177) is ${}^{4B}\!/\!{}_{\xi}\sqrt{{}^{1}\!/\!{}_{n}}$. When we reduce this term, we get a tighter upper bound for linear probing. As a result, in the SSL setup, linear probing performed on top of CL features results in a predictor with a much smaller value of the upper bound, when compared with linear probing done on inputs directly. Even for larger k, as long as $k = o(d)$ the generalization error bound for the CL predictor under the SSL setup reduces drastically compared to ERM. This explains why doing further self-training over the CL predictor in the SSL setup does not result in big gains on the target accuracy as compared to the UDA setting.

E.6 Limitations of Prior Work

E.6.1 Contrastive learning analysis

Prior works that analyze contrastive learning show that minimizers of the CL objective recover clusters in the augmentation graph, which weights pairs of augmentations with their probability of being sampled as a positive pair (Cabannes et al., 2023; HaoChen et al., 2021; Johnson et al., 2022; Saunshi et al., 2022). When there is no distribution shift in the downstream task, assumptions made on the graph in the form of consistency of augmentations with downstream labels, is sufficient to ensure that a linear probed head has good ID generalization. Under distribution shift, these assumptions are not sufficient and stronger ones are needed. *E.g.*, some works assume that same-domain/class examples are weighted higher that cross-class cross-domain pairs (HaoChen et al., 2022; Shen et al., 2022).

Using notation defined in (Shen et al., 2022), the assumption on the augmentation graph requires cross-class and same-domain weights (β) to be higher than cross-class and cross-domain weights (γ). It is unclear if examples from different classes in the same domain will be "connected" if strong spurious features exist in the source domain and augmentations fail to mask them completely (*e.g.*, image background may not be completely masked by augmentations but it maybe perfectly predictive of the label on source domain). In such cases, the linear predictor learnt over CL would fail to generalize OOD. In our toy setup as well, the connectivity assumption fails since on source x_{sp} is perfectly predictive of the label and the augmentations are imperfect, *i.e.*, augmentations do not mask x_{sp} and examples of different classes do not overlap in source (*i.e.*, $\beta = 0$). On the other hand, since x_{sp} is now random on target, augmentations of different classes may overlap, *i.e.*, $\gamma > 0$, thus breaking the connectivity assumption. This is also highlighted in our empirical findings of CL furnishing representations that do not fully enable linear transferability from source to target (see Sec. 7.5). These empirical findings also call into question existing assumptions on data augmentations, highlighting that perfect linear transferability may not typically hold in practice. It is in this setting that we believe self-training can improve over contrastive learning by unlearning source-only features and improving linear transferability.

E.6.2 Self-training analysis

Some prior works on self-training view it as consistency regularization that constrain pseudolabels of original samples to be consistent with all their augmentations (Cai et al., 2021a; Sohn et al., 2020; Wei et al., 2020). This framework abstracts the role played by the optimization algorithm and instead evaluates the global minimizer of a population objective that enforces consistency of pseudolabels. In addition, certain expansion assumptions on class-conditional distributions are needed to ensure that pseudolabels have good accuracy on source and target domains. This framework does not account for challenges involved in propagating labels iteratively. For *e.g.*, when augmentation distribution has long tails, the consistency of pseudolabels depends on the sampling frequency of "favorable" augmentations. As an illustration, consider our augmentation distribution in the toy setup in Sec. 12.4. If it were not uniform over dimensions, but instead something that was highly skewed, then a large number of augmentations need to be sampled for every data point to propagate pseudolabels successfully from source labeled samples to target unlabeled samples during self-training. This might hurt the performance of ST when we are optimizing for only finitely many iterations and over finitely many datapoints. This is why in our analysis we instead adopt the iterative analysis of self-training (Chen et al., 2020b).

E.7 Additional Lemmas

In this section we define some additional lemmas that we use in our theoretical analysis in E.5.

Lemma E.7.1 (Upper bound and lower bounds on erfc; Kschischang (2017))**.** *Define* $\operatorname{erfc}(x) = \frac{2}{\sqrt{\pi}} \cdot \int_x^\infty \exp(-z^2) \cdot dz$*. Then we have:*

$$\frac{2}{\sqrt{\pi}} \cdot \frac{\exp(-x^2)}{x + \sqrt{x^2 + 2}} < \operatorname{erfc}(x) \leqslant \frac{2}{\sqrt{\pi}} \cdot \frac{\exp(-x^2)}{x + \sqrt{x^2 + 4/\pi}}$$

Lemma E.7.2 (Properties of Mill's ratio (Baricz, 2008))**.** *Define the Mill's ratio as* $\mathrm{r}\left(x\right) = \exp\left(x^2/2\right) \cdot \operatorname{erfc}\left(x/\sqrt{2}\right) \cdot \sqrt{\pi/2}$*. Then following assertions are true: (i)* $\mathrm{r}\left(x\right)$ *is a strictly decreasing log-convex function; (ii)* $\mathrm{r}'(x) = x \cdot \mathrm{r}\left(x\right) - 1$ *is an increasing function with* $\mathrm{r}'(x) < 0$ *for all x; (iii)* $\mathrm{r}''(x) = \mathrm{r}\left(x\right) + x^2 \cdot \mathrm{r}\left(x\right) - x$ *is a decreasing function with* $\mathrm{r}''(x) > 0$ *for all x; (iv)* $x^2 \cdot \mathrm{r}'(x)$ *is a decreasing function of x.*

Lemma E.7.3 (invariance loss as product with operator L)**.** *The invariance loss for some $\phi \in \mathbb{R}^d$ is given as: $2 \cdot \int_{\mathcal{A}} \phi(a) \cdot L(\phi)(a) \, \mathrm{dP}_{\mathsf{A}}$ where the operator L is defined as:*

$$L(\phi)(a) = \phi(a) - \int_{\mathcal{A}} \frac{A_+(a, a')}{p_{\mathsf{A}}(a)} \cdot \phi(a') \, \mathrm{d}a'$$

Proof. The invariance loss for ϕ is given by:

$$\mathbb{E}_{x \sim \mathrm{P}_{\mathsf{U}}} \mathbb{E}_{a_1, a_2 \sim \mathrm{P}_{\mathsf{A}}(\cdot|x)} (a_1^\top \phi - a_2^\top \phi)^2 = 2\mathbb{E}_{x \sim \mathrm{P}_{\mathsf{U}}} \mathbb{E}_{a \sim \mathrm{P}_{\mathsf{A}}(\cdot|x)} \left[\phi(a)^2\right] - 2\mathbb{E}_{a_1, a_2 \sim A_+(\cdot,\cdot)} \left[\phi(a_1)\phi(a_2)\right] \tag{E.178}$$

$$= 2 \cdot \int_{\mathcal{A}} \phi(a)^2 \, \mathrm{dP}_{\mathsf{A}} - 2 \cdot \int_{\mathcal{A}} \phi(a) \left(\int_{\mathcal{A}} \frac{A_+(a, a_2)}{p_{\mathsf{A}}(a)} \cdot \phi(a_2) \, \mathrm{d}a_2 \right) \mathrm{dP}_{\mathsf{A}} \tag{E.179}$$

$$= 2 \cdot \int_{\mathcal{A}} \phi(a) \cdot L(\phi)(a) \, \mathrm{dP}_{\mathsf{A}} \tag{E.180}$$

□

Lemma E.7.4. *If $\mathcal{W}$ is the space spanned by w_{in} and w_{sp}, and $\mathcal{W}_\perp$ is the null space for $\mathcal{W}$, then for any $u \in \mathcal{W}$ and any $v \in \mathcal{W}_\perp$, the covariance along these directions $\mathbb{E}_{a \sim \mathrm{P}_{\mathsf{A}}}[a^\top u v^\top a] = 0$.*

Proof: We can write the covariance over augmentations after we break down the augmentation a into two projections: $a = \Pi_{\mathcal{W}}(a) + \Pi_{\mathcal{W}_\perp}(a)$

$$\mathbb{E}_{a \sim \mathrm{P}_{\mathsf{A}}}[a^\top u v^\top a] = \mathbb{E}_{a \sim \mathrm{P}_{\mathsf{A}}} \left[\left(u^\top(\Pi_{\mathcal{W}}(a) + \Pi_{\mathcal{W}_\perp}(a))\right) \left(v^\top(\Pi_{\mathcal{W}}(a) + \Pi_{\mathcal{W}_\perp}(a))\right) \right] \tag{E.181}$$

$$= \mathbb{E}_{a \sim \mathrm{P}_{\mathsf{A}}} \left[\left(u^\top \Pi_{\mathcal{W}}(a)\right) \left(v^\top \Pi_{\mathcal{W}_\perp}(a)\right) \right] \tag{E.182}$$

$$= u^\top \left(\mathbb{E}_{a \sim \mathrm{P}_{\mathsf{A}}} \left[\Pi_{\mathcal{W}}(a) \Pi_{\mathcal{W}_\perp}(a)^\top \right] \right) v = 0 \tag{E.183}$$

where the last inequality follows from the fact that $\mathbb{E}_{a \sim \mathrm{P}_{\mathsf{A}}} \left[\Pi_{\mathcal{W}}(a) \Pi_{\mathcal{W}_\perp}(a)^\top\right] = \mathbb{E}_{a \sim \mathrm{P}_{\mathsf{A}}} \left[\Pi_{\mathcal{W}}(a)\right] \mathbb{E}_{a \sim \mathrm{P}_{\mathsf{A}}} \left[\Pi_{\mathcal{W}_\perp}(a)\right]^\top$, since the noise in the null space of $\mathcal{W}$ is drawn independent of the component along $\mathcal{W}$, and furthermore the individual expectations evaluate to zero.

Lemma E.7.5 (closed-form expressions for eigenvalues and eigenvectors of $\Sigma_A, \widetilde{\Sigma}$)**.** *For a 2×2 real symmetric matrix $\begin{bmatrix} a, & b \\ c, & d \end{bmatrix}$ the eigenvalues λ_1, λ_2 are given by the following expressions:*

$$\lambda_1 = \frac{(a + b + \delta)}{2}, \quad \lambda_2 = \frac{(a + b - \delta)}{2},$$

where $\delta = \sqrt{4c^2 + (a - b)^2}$. Further, the eigenvectors are given by $U = \begin{bmatrix} \cos(\theta), & \sin(\theta) \\ \sin(\theta), & -cos(\theta) \end{bmatrix}$, where:

$$\tan(\theta) = \frac{b - a + \delta}{2c}.$$

For full proof of these statements see (Deledalle et al., 2017). Here, we will use these statements to arrive at closed form expressions for the eigenvalues and eigenvectors of Σ_A, $\widetilde{\Sigma}$.

Proof. We can now substitute the above formulae with a, b, c, d taken from the expressions of Σ_A and $\widetilde{\Sigma}$, to get the following values: λ_1, λ_2 are the eigenvalues of Σ_A, with α determining the corresponding eigenvectors $[\cos(\alpha), \sin(\alpha)], [\sin(\alpha), -\cos(\alpha)]$; and $\widetilde{\lambda}_1, \widetilde{\lambda}_2$ are the eigenvalues of $\widetilde{\Sigma}$, with β determining the corresponding eigenvectors: $[\cos(\beta), \sin(\beta)], [\sin(\beta), -\cos(\beta)]$.

$$\begin{aligned}\lambda_1 = \frac{1}{8}\Bigg(&\gamma^2\left(1+\frac{1}{3d_{\text{in}}}\right)+\frac{\sigma_{\text{in}}^2}{3}\left(1-\frac{1}{d_{\text{in}}}\right)+\frac{d_{\text{sp}}}{2}+\frac{2\sigma_{\text{sp}}^2}{3}+\frac{1}{6}\\ &+\sqrt{\gamma^2 d_{\text{sp}}+\left(\left(\gamma^2\left(1+\frac{1}{3d_{\text{in}}}\right)+\frac{\sigma_{\text{in}}^2}{3}\left(1-\frac{1}{d_{\text{in}}}\right)\right)-\left(\frac{d_{\text{sp}}}{2}+\frac{2\sigma_{\text{sp}}^2}{3}+\frac{1}{6}\right)\right)^2}\Bigg)\end{aligned} \tag{E.184}$$

$$\begin{aligned}\lambda_2 = \frac{1}{8}\Bigg(&\gamma^2\left(1+\frac{1}{3d_{\text{in}}}\right)+\frac{\sigma_{\text{in}}^2}{3}\left(1-\frac{1}{d_{\text{in}}}\right)+\frac{d_{\text{sp}}}{2}+\frac{2\sigma_{\text{sp}}^2}{3}+\frac{1}{6}\\ &-\sqrt{\gamma^2 d_{\text{sp}}+\left(\left(\gamma^2\left(1+\frac{1}{3d_{\text{in}}}\right)+\frac{\sigma_{\text{in}}^2}{3}\left(1-\frac{1}{d_{\text{in}}}\right)\right)-\left(\frac{d_{\text{sp}}}{2}+\frac{2\sigma_{\text{sp}}^2}{3}+\frac{1}{6}\right)\right)^2}\Bigg)\end{aligned} \tag{E.185}$$

$$\widetilde{\lambda}_1 = \frac{1}{8}\left(\gamma^2+\frac{d_{\text{sp}}}{2}+\frac{\sigma_{\text{sp}}^2}{2}+\sqrt{\gamma^2 d_{\text{sp}}+\left(\gamma^2-\left(\frac{d_{\text{sp}}}{2}+\frac{\sigma_{\text{sp}}^2}{2}\right)\right)^2}\right) \tag{E.186}$$

$$\widetilde{\lambda}_2 = \frac{1}{8}\left(\gamma^2+\frac{d_{\text{sp}}}{2}+\frac{\sigma_{\text{sp}}^2}{2}-\sqrt{\gamma^2 d_{\text{sp}}+\left(\gamma^2-\left(\frac{d_{\text{sp}}}{2}+\frac{\sigma_{\text{sp}}^2}{2}\right)\right)^2}\right) \tag{E.187}$$

$$\begin{aligned}\tan(\alpha) = \frac{1}{\gamma\sqrt{d_{\text{sp}}}}\Bigg(&\frac{d_{\text{sp}}}{2}+\frac{2\sigma_{\text{sp}}^2}{3}+\frac{1}{6}-\left(\gamma^2\left(1+\frac{1}{3d_{\text{in}}}\right)+\frac{\sigma_{\text{in}}^2}{3}\left(1-\frac{1}{d_{\text{in}}}\right)\right)\\ &+\sqrt{\gamma^2 d_{\text{sp}}+\left(\left(\gamma^2\left(1+\frac{1}{3d_{\text{in}}}\right)+\frac{\sigma_{\text{in}}^2}{3}\left(1-\frac{1}{d_{\text{in}}}\right)\right)-\left(\frac{d_{\text{sp}}}{2}+\frac{2\sigma_{\text{sp}}^2}{3}+\frac{1}{6}\right)\right)^2}\Bigg)\end{aligned} \tag{E.188}$$

$$\tan(\beta) = \frac{1}{\gamma\sqrt{d_{\text{sp}}}}\left(\frac{d_{\text{sp}}}{2}+\frac{\sigma_{\text{sp}}^2}{2}-\gamma^2+\sqrt{\gamma^2 d_{\text{sp}}+\left(\gamma^2-\left(\frac{d_{\text{sp}}}{2}+\frac{\sigma_{\text{sp}}^2}{2}\right)\right)^2}\right) \tag{E.189}$$

Consider the subclass of problem parameters, $d_{\text{sp}} = z, \gamma = {}^{K_1}\!/\!\sqrt{z}$ and $\sigma_{\text{sp}} = K_2\sqrt{z}$ for fixed constants $K_1, K_2 > 0$ and some variable $z > 0$, which we can vary to give us different problem instances for our toy model in (E.3).

$$\lambda_1 = \frac{1}{8}\Bigg(\frac{K_1^2}{z}\left(1+\frac{1}{3d_{\text{in}}}\right)+\frac{\sigma_{\text{in}}^2}{3}\left(1-\frac{1}{d_{\text{in}}}\right)+\frac{z}{2}+\frac{2K_2^2 z}{3}+\frac{1}{6}$$

$$+\sqrt{K_1^2+\left(\left(\frac{K_1^2}{z}\left(1+\frac{1}{3d_{\text{in}}}\right)+\frac{\sigma_{\text{in}}^2}{3}\left(1-\frac{1}{d_{\text{in}}}\right)\right)-\left(\frac{z}{2}+\frac{2K_2^2z}{3}+\frac{1}{6}\right)\right)^2}\Bigg) \tag{E.190}$$

$$\begin{aligned}\lambda_2=\frac{1}{8}\Bigg(&\frac{K_1^2}{z}\left(1+\frac{1}{3d_{\text{in}}}\right)+\frac{\sigma_{\text{in}}^2}{3}\left(1-\frac{1}{d_{\text{in}}}\right)+\frac{z}{2}+\frac{2K_2^2z}{3}+\frac{1}{6}\\&-\sqrt{K_1^2+\left(\left(\frac{K_1^2}{z}\left(1+\frac{1}{3d_{\text{in}}}\right)+\frac{\sigma_{\text{in}}^2}{3}\left(1-\frac{1}{d_{\text{in}}}\right)\right)-\left(\frac{z}{2}+\frac{2K_2^2z}{3}+\frac{1}{6}\right)\right)^2}\Bigg)\end{aligned} \tag{E.191}$$

$$\widetilde{\lambda}_1=\frac{1}{8}\left(\frac{K_1^2}{z}+\frac{z}{2}+\frac{K_2^2z}{2}+\sqrt{K_1^2+\left(\frac{K_1^2}{z}-\left(\frac{z}{2}+\frac{K_2^2z}{2}\right)\right)^2}\right) \tag{E.192}$$

$$\widetilde{\lambda}_2=\frac{1}{8}\left(\frac{K_1^2}{z}+\frac{z}{2}+\frac{K_2^2z}{2}-\sqrt{K_1^2+\left(\frac{K_1^2}{z}-\left(\frac{z}{2}+\frac{K_2^2z}{2}\right)\right)^2}\right) \tag{E.193}$$

$$\begin{aligned}\tan(\alpha)=\frac{1}{K_1}\Bigg(&\frac{z}{2}+\frac{2K_2^2z}{3}+\frac{1}{6}-\left(\frac{K_1^2}{z}\left(1+\frac{1}{3d_{\text{in}}}\right)+\frac{\sigma_{\text{in}}^2}{3}\left(1-\frac{1}{d_{\text{in}}}\right)\right)\\&+\sqrt{K_1^2+\left(\frac{K_1^2}{z}\left(1+\frac{1}{3d_{\text{in}}}\right)+\frac{\sigma_{\text{in}}^2}{3}\left(1-\frac{1}{d_{\text{in}}}\right)-\left(\frac{z}{2}+\frac{2K_2^2z}{3}+\frac{1}{6}\right)\right)^2}\Bigg)\end{aligned} \tag{E.194}$$

$$\tan(\beta)=\frac{1}{K_1}\left(\frac{z}{2}+\frac{K_2^2z}{2}-\frac{K_1^2}{z}+\sqrt{K_1^2+\left(\frac{K_1^2}{z}-\left(\frac{z}{2}+\frac{K_2^2z}{2}\right)\right)^2}\right) \tag{E.195}$$

From Stewart (1993), we can use the closed form expression for the singular vectors of a 2×2 full rank asymmetric matrix $\begin{bmatrix}a, & b\\ c, & d\end{bmatrix}$. The singular vectors are given by

$$\begin{bmatrix}\cos\theta, & \sin\theta\\ \sin\theta, & -\cos\theta\end{bmatrix},$$

where, $\tan(2\theta)$ is given by:

$$\tan(2\theta)=\frac{2ac+2bd}{a^2+b^2-c^2-d^2}.$$

Now, substituting the values in the expression from (E.97), we get singular vectors of the above form where $\theta\in[0,\pi/2]$ satisfies:

$$\theta = \frac{1}{2}\tan^{-1}\left(\frac{2\tan(\beta-\alpha);\cdot(\widetilde{\lambda}_1-\widetilde{\lambda}_2)\cdot\sqrt{\lambda_1\lambda_2}}{(\lambda_2\widetilde{\lambda}_1-\lambda_1\widetilde{\lambda}_2)-(\lambda_1\widetilde{\lambda}_1-\lambda_2\widetilde{\lambda}_2)\cdot\tan^2(\alpha-\beta)}\right) \tag{E.196}$$

□

Lemma E.7.6 (asymptotic behavior of $\tau\tan\theta$)**.** *For $\gamma = {}^{K_1}\!/\!_{\sqrt{z}}$, $\sigma_{\text{sp}} = K_2\sqrt{z}$,*

$$\lim_{z\to\infty}\tau\tan\theta = \frac{K_1K_2^2}{(1+K_2^2)2\sigma_{\text{in}}^2(1-{}^1\!/\!_{d_{\text{in}}})}$$

Proof. In order to determine the asymptotic nature of $\tan(\theta)$ as $z\to\infty$, we take the limit of a slightly different term first, since we have the closed form expression of $\tan(2\theta)$.

$$\begin{aligned}\lim_{z\to\infty}\tau\tan(2\theta) &= \sqrt{\frac{\lambda_1}{\lambda_2}}\cdot\frac{2\tan(\alpha-\beta)\cdot({}^{\tilde{\lambda}_1}\!/\!_{\widetilde{\lambda}_2}-1)}{({}^{\tilde{\lambda}_1}\!/\!_{\tilde{\lambda}_2}-{}^{\lambda_1}\!/\!_{\lambda_2})-({}^{\lambda_1\tilde{\lambda}_1}\!/\!_{\lambda_2\tilde{\lambda}_2}-1)\cdot\tan^2(\alpha-\beta)}\\ &= 2\tan(\alpha-\beta)\cdot\frac{{}^{\widetilde{\lambda}_1}\!/\!_{\widetilde{\lambda}_2}-1}{{}^{\widetilde{\lambda}_1}\!/\!_{\widetilde{\lambda}_2}\cdot{}^{\lambda_2}\!/\!_{\lambda_1}-1},\end{aligned}$$

since it is easy to see that $\lim_{z\to\infty}\tan^2(\alpha-\beta)\cdot\left(\frac{\lambda_1\widetilde{\lambda}_1}{\lambda_2\widetilde{\lambda}_2}-1\right)=0$.

If we use $\tan(\alpha-\beta)=\frac{\tan\alpha-\tan\beta}{1+\tan\alpha\tan\beta}$, and substitute the functions of z, for all the quantities in the above expression using Lemma E.7.5, we derive: $\lim_{z\to\infty}\tau\tan 2\theta = {}^{2K_1K_2^2}\!/\!_{(1+K_2^2)2\sigma_{\text{in}}^2(1-{}^1\!/\!_{d_{\text{in}}})}$.

Since $\tau\to\infty$, $\tan(2\theta)\to 0$, and further from Taylor approximation of $\tan(2\theta)$, $\tan(2\theta)\to 2\theta$. We can use this to derive the limit for $\tau\tan\theta$, which would just be ${}^1\!/\!_2\cdot{}^{2K_1K_2^2}\!/\!_{(1+K_2^2)2\sigma_{\text{in}}^2(1-{}^1\!/\!_{d_{\text{in}}})} = {}^{K_1K_2^2}\!/\!_{(1+K_2^2)2\sigma_{\text{in}}^2(1-{}^1\!/\!_{d_{\text{in}}})}$.

□

Lemma E.7.7 (asymptotic behaviors of $\cot\alpha,\tan\theta$)**.** *For $\gamma = {}^{K_1}\!/\!_{\sqrt{z}}$, $\sigma_{\text{sp}} = K_2\sqrt{z}$ following the expressions in Lemma E.7.5,*

$$\lim_{z\to\infty}\cot\alpha = 0, \qquad \lim_{z\to\infty}\tan\theta = 0.$$

Proof. For $\tan\theta$, since $\tau\to\infty$, and $\tau\tan\theta$ approaches a constant (from Lemma E.7.6), we conclude $\lim_{z\to\infty}\tan\theta = 0$. For $\cot\alpha$,

$$\lim_{z\to\infty}\frac{z}{2}+\frac{2K_2^2z}{3}+\frac{1}{6}-\left(\frac{K_1^2}{z}\left(1+\frac{1}{3d_{\text{in}}}\right)+\frac{\sigma_{\text{in}}^2}{3}\left(1-\frac{1}{d_{\text{in}}}\right)\right)=\infty,$$

and,

$$\lim_{z\to\infty}\sqrt{K_1^2+\left(\frac{K_1^2}{z}\left(1+\frac{1}{3d_{\text{in}}}\right)+\frac{\sigma_{\text{in}}^2}{3}\left(1-\frac{1}{d_{\text{in}}}\right)-\left(\frac{z}{2}+\frac{2K_2^2z}{3}+\frac{1}{6}\right)\right)^2}=\infty.$$

Thus, $\cot\alpha\to 0$. □

Lemma E.7.8 (asymptotic behavior of $z \cot \alpha$)**.** *For* $\gamma = {}^{K_1}\!/\!\sqrt{z}$, $\sigma_{\text{sp}} = K_2\sqrt{z}$ *following the expressions in Lemma E.7.5,*

$$\lim_{z \to \infty} z \cot \alpha = \frac{K_1}{1 + {}^{4}\!/\!{}_{3}K_2^2}.$$

Proof. The expression for $z \cot \alpha$ or ${}^{z}\!/\!{}_{\tan \alpha}$ follows from Lemma E.7.5:

$$\lim_{z \to \infty} z \cot \alpha = \frac{zK_1}{p + \sqrt{p^2 + K_1^2}},$$

where $p = \frac{K_1^2}{z}\left(1 + \frac{1}{3d_{\text{in}}}\right) + \frac{\sigma_{\text{in}}^2}{3}\left(1 - \frac{1}{d_{\text{in}}}\right) - \left(\frac{z}{2} + \frac{2K_2^2 z}{3} + \frac{1}{6}\right)$. Applying L'Hôpital's (relevant expressions are continuous in z) rule we get: $\lim_{z \to \infty} z \cot \alpha = \frac{K_1}{1 + {}^{4}\!/\!{}_{3}K_2^2}$.

□

Lemma E.7.9 (asymptotic behavior of ${}^{z}\!/\!{}_{\tau^2}$)**.** *For* $\gamma = {}^{K_1}\!/\!\sqrt{z}$, $\sigma_{\text{sp}} = K_2\sqrt{z}$ *following the expressions in Lemma E.7.5,*

$$\lim_{z \to \infty} {}^{z}\!/\!{}_{\tau^2} = \frac{2\sigma_{\text{in}}^2/3(1 - {}^{1}\!/\!{}_{d_{\text{in}}})}{1 + {}^{4}\!/\!{}_{3}K_2^2}.$$

Proof. For $\tau = {}^{\lambda_1}\!/\!{}_{\lambda_2}$, substituting the relevant expressions from Lemma E.7.5, we get:

$$\begin{aligned} {}^{z}\!/\!{}_{\tau^2} &= \frac{z\lambda_2}{\lambda_1} \\ &= z \cdot \frac{{}^{2K_1^2}\!/\!{}_{z}\left(1 + {}^{1}\!/\!{}_{3d_{\text{in}}}\right) + 2\sigma_{\text{in}}^2\left(1 - {}^{1}\!/\!{}_{d_{\text{in}}}\right) + p - \sqrt{K_1^2 + p^2}}{{}^{2K_1^2}\!/\!{}_{z}\left(1 + {}^{1}\!/\!{}_{3d_{\text{in}}}\right) + 2\sigma_{\text{in}}^2\left(1 - {}^{1}\!/\!{}_{d_{\text{in}}}\right) + p + \sqrt{K_1^2 + p^2}}, \end{aligned}$$

where $p = {}^{z}\!/\!{}_{2} + {}^{2K_2^2 z}\!/\!{}_{3} + {}^{1}\!/\!{}_{6}$. Applying L'Hôpital's (relevant expressions are continuous in z) rule we get: $\lim_{z \to \infty} {}^{z}\!/\!{}_{\tau^2} = \frac{2\sigma_{\text{in}}^2/3(1 - {}^{1}\!/\!{}_{d_{\text{in}}})}{1 + {}^{4}\!/\!{}_{3}K_2^2}$. □

Lemma E.7.10 (0-1 error of a classifier on target)**.** *Assume a classifier of the form* $w = l_1 \cdot w_{\text{in}} + l_2 \cdot w_{\text{sp}}$ *where* $l_1, l_2 \in \mathbb{R}$ *and* $w_{\text{in}} = [w^\star, 0, ..., 0]^\top$, *and* $w_{\text{sp}} = [0, ..., 0, {}^{\mathbf{1}_{d_{\text{sp}}}}\!/\!\sqrt{d_{\text{sp}}}]^\top$. *Then the target accuracy of this classifier is given by* $0.5 \cdot \operatorname{erfc}\left(-\frac{l_1 \cdot \gamma}{\sqrt{2} \cdot l_2 \cdot \sigma_{\text{sp}}}\right)$.

Proof. Assume $(x, y) \sim \mathrm{P}_t$. Accuracy of w is given by $\mathbb{E}_{\mathrm{P}_t}\left[\left(\text{sign}\left(w^\top x\right) = y\right)\right]$.

$$\begin{aligned} \mathbb{E}_{\mathrm{P}_t}\left[\text{sign}\left(w^\top x\right) = y\right] &= \mathbb{E}_{\mathrm{P}_t}\left[y \cdot \text{sign}\left(w^\top x\right) = 1\right] \\ &= \mathbb{E}_{\mathrm{P}_t}\left[y \cdot \left(w^\top x\right) > 0\right] \\ &= \mathbb{E}_{\mathrm{P}_t}\left[y \cdot \left(x^\top (l_1 \cdot w_{\text{in}} + l_2 \cdot w_{\text{sp}})\right) > 0\right] \\ &= \mathbb{E}_{\mathrm{P}_t}\left[y \cdot \left(\gamma \cdot l_1 \cdot y + l_2 \cdot \sigma_{\text{sp}}\right) > 0\right] \\ &= \mathbb{E}_{z \sim \mathcal{N}(0,1)}\left[\left(\gamma \cdot l_1 + y \cdot l_2 \cdot \sigma_{\text{sp}} \cdot z\right) > 0\right] \\ &= \mathbb{E}_{z \sim \mathcal{N}(0,1)}\left[y \cdot l_2 \cdot \sigma_{\text{sp}} \cdot z > -\gamma \cdot l_1\right] \end{aligned}$$

$$
\begin{aligned}
&= \mathbb{E}_{z \sim \mathcal{N}(0,1)} \left[l_2 \cdot \sigma_{\text{sp}} \cdot z > -\gamma \cdot l_1 \right] \\
&= \mathbb{E}_{z \sim \mathcal{N}(0,1)} \left[z > -\frac{\gamma \cdot l_1}{l_2 \cdot \sigma_{\text{sp}}} \right]
\end{aligned}
$$

Using the definition of erfc function, we get the aforementioned accuracy expression. □

Lemma E.7.11. *For $\sigma > 0$ and $\mu \in \mathbb{R}$, we have*

$$
\begin{aligned}
g(\mu, \sigma) &:= \mathbb{E}_{z \sim \mathcal{N}(0,\sigma)} \left[\exp\left(-\left|\mu + z\right|\right) \right] && \text{(E.197)} \\
&= \frac{1}{2} \left(\exp\left({}^{\sigma^2}\!/_2 - \mu \right) \cdot \operatorname{erfc}\left(-{}^{\mu}\!/_{\sqrt{2}\sigma} + {}^{\sigma}\!/_{\sqrt{2}} \right) + \exp\left({}^{\sigma^2}\!/_2 + \mu \right) \cdot \operatorname{erfc}\left({}^{\mu}\!/_{\sqrt{2}\sigma} + {}^{\sigma}\!/_{\sqrt{2}} \right) \right) && \text{(E.198)}
\end{aligned}
$$

Proof. The proof uses simple algebra and the definition of erfc function.

$$
\begin{aligned}
g(\mu, \sigma) &:= \mathbb{E}_{z \sim \mathcal{N}(0,\sigma)} \left[\exp\left(-\left|\mu + z\right|\right) \right] \\
&= \frac{1}{\sqrt{2\pi}} \int_z \exp\left(-\left|\mu + z\right|\right) \cdot \exp\left(-\frac{z^2}{2\sigma^2}\right) dz \\
&= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \exp\left(-\left|\mu + z\right|\right) \cdot \exp\left(-\frac{z^2}{2\sigma^2}\right) dz \\
&= \frac{1}{\sqrt{2\pi}} \int_{-\mu}^{\infty} \exp\left(-\mu + z\right) \cdot \exp\left(-\frac{z^2}{2\sigma^2}\right) dz + \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{-\mu} \exp\left(\mu + z\right) \cdot \exp\left(-\frac{z^2}{2\sigma^2}\right) dz \\
&= \exp\left(\sigma^2/2 - \mu\right) \int_{\frac{-\mu}{\sqrt{2}\sigma} + \frac{\sqrt{2}\sigma}{2}}^{\infty} \exp(-z^2) dz + \exp\left(\sigma^2/2 + \mu\right) \int_{-\infty}^{\frac{-\mu}{\sqrt{2}\sigma} - \frac{\sqrt{2}\sigma}{2}} \exp(-z^2) dz \\
&= \frac{1}{2} \left(\exp\left({}^{\sigma^2}\!/_2 - \mu \right) \cdot \operatorname{erfc}\left(-{}^{\mu}\!/_{\sqrt{2}\sigma} + {}^{\sigma}\!/_{\sqrt{2}} \right) + \exp\left({}^{\sigma^2}\!/_2 + \mu \right) \cdot \operatorname{erfc}\left({}^{\mu}\!/_{\sqrt{2}\sigma} + {}^{\sigma}\!/_{\sqrt{2}} \right) \right)
\end{aligned}
$$

□

Appendix F

Appendix: RLSbench: Domain Adaptation Under Relaxed Label Shift

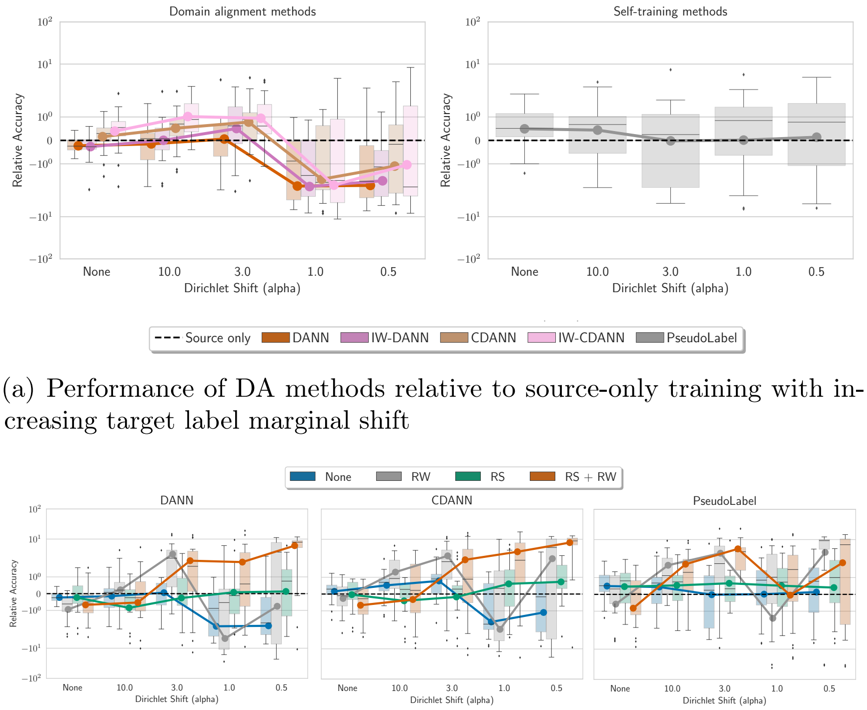
F.1 Description of Plots

For each plot in Fig. 7.2, we obtain all the distribution shift pairs with a specific alpha (i.e., the value on the x-axis). Then for each distribution shift pair (with a specific alpha value), we obtain *relative performance* by subtracting the performance of a source-only model trained on the source dataset of that distribution shift pair from the performance of the model trained on that distribution shift pair with the DA algorithm of interest. Thus for each alpha and each DA method, we obtain 112 relative performance values. We draw the box plot and the mean of these relative performance values.

For (similar-looking) plots, we use the same technique throughout the chapter. The only thing that changes is the group of points over which aggregation is performed.

F.2 Tabular and NLP Results Omitted from the Main Paper

F.2.1 Tabular Datasets

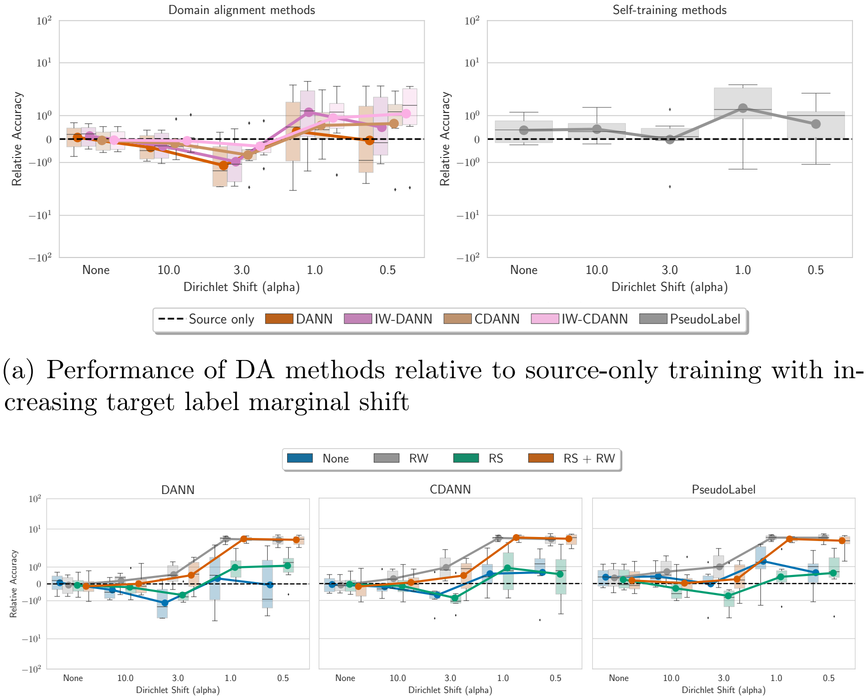


(a) Performance of DA methods relative to source-only training with increasing target label marginal shift

(b) Relative performance of DA methods when paired with our meta-algorithm (RS and RW corrections)

Figure F.1: *Performance of different DA methods relative to a source-only model across all distribution shift pairs in tabular datasets grouped by shift severity in label marginal.* For each distribution shift pair and DA method, we plot the relative accuracy of the model trained with that DA method by subtracting the accuracy of the source-only model. Hence, the black dotted line at 0 captures the performance of the source-only model. Smaller the Dirichlet shift parameter, the more severe is the shift in target class proportion. **(a)** Shifts with $\alpha = \{\textsc{None}, 10.0, 3.0\}$ have little to no impact on different DA methods whereas the performance of all DA methods degrades when $\alpha \in \{1.0, 0.5\}$ often falling below the performance of a source-only classifier. **(b)** RS and RW (in our meta-algorithm) together significantly improve aggregate performance over no correction for all DA methods. While RS consistently helps (over no correction) across different label marginal shift severities, RW hurts slightly when shift severity is small. However, for severe shifts ($\alpha \in \{3.0, 1.0, 0.5\}$) RW significantly improves performance for all the methods.

F.2.2 NLP Datasets



(a) Performance of DA methods relative to source-only training with increasing target label marginal shift

(b) Relative performance of DA methods when paired with our meta-algorithm (RS and RW corrections)

Figure F.2: *Performance of different DA methods relative to a source-only model across all distribution shift pairs in NLP datasets grouped by shift severity in label marginal.* For each distribution shift pair and DA method, we plot the relative accuracy of the model trained with that DA method by subtracting the accuracy of the source-only model. Hence, the black dotted line at 0 captures the performance of the source-only model. Smaller the Dirichlet shift parameter, the more severe is the shift in target class proportion. **(a)** Performance of DANN and IW-DANN methods degrades with increasing severity of target label marginal shift often falling below the performance of a source-only classifier (except for Noisy Student). Performance of PsuedoLabel, CDANN, and IW-CDANN show less susceptibility to increasing severity in target marginal shift. **(b)** RS and RW (in our meta-algorithm) together significantly improve aggregate performance over no correction for all DA methods. While RS consistently helps (over no correction) across different label marginal shift severities, RW hurts slightly for BN-adapt, TENT, and NoisyStudent when shift severity is small. However, for severe shifts ($\alpha \in \{3.0, 1.0, 0.5\}$) RW significantly improves performance for all the methods. Detailed results with all methods on individual datasets in App. **??**.

F.3 Comparison between IW-CDANN, IW-DANN, and SENTRY with Existing DA methods paired with our Meta-Algorithm

Fig. F.3 shows the relevant comparison.

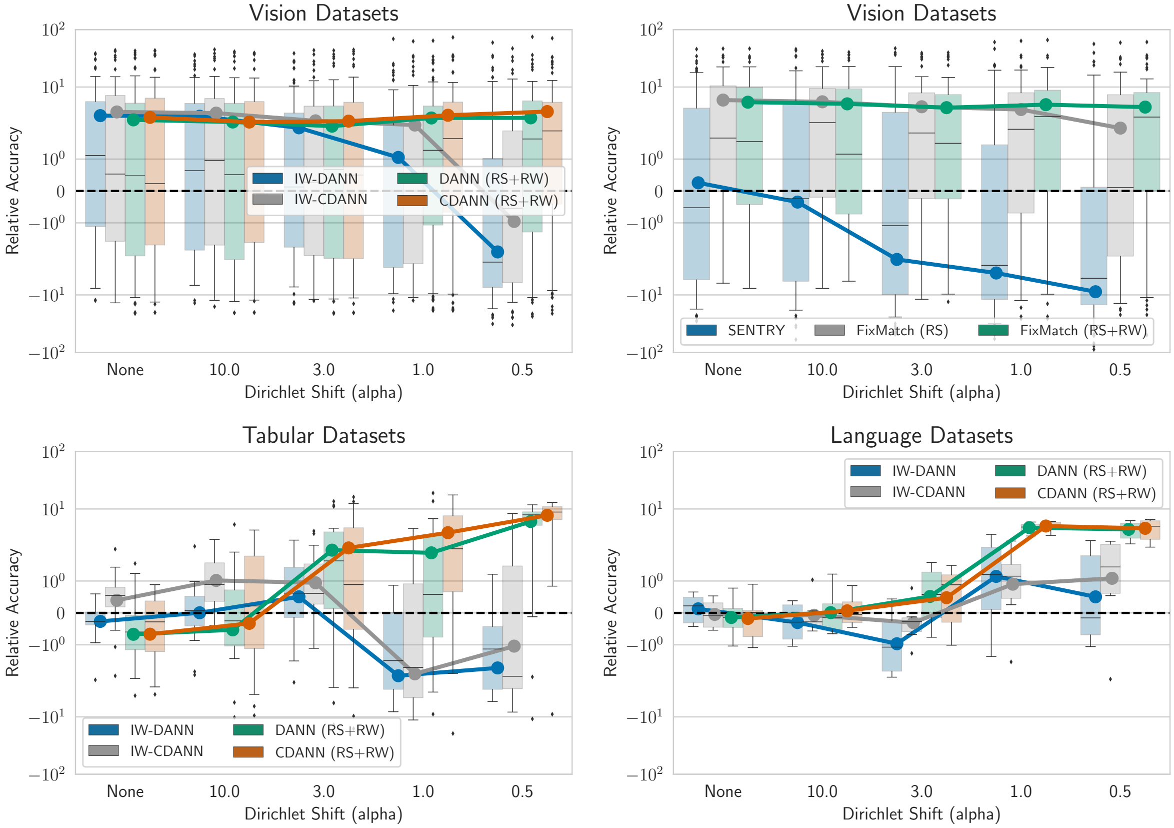


Figure F.3: *Comparison of existing DA methods paired with our RS and RW correction and DA methods specifically proposed for relaxed label shift problems.* Across vision and tabular datasets, we observe the susceptibility of IW-DAN, IW-CDAN, and SENTRY with increasing severity of target label marginal shifts. In particular, for severe target label marginal shifts, the performance of IW-DAN, IW-CDAN, and SENTRY often falls below that of the source-only model. However, existing DA techniques when paired with RS + RW correction significantly improve over the source-only model. For NLP, datasets we observe similar behavior but with relatively less intensity.

Note. On Officehome dataset, we observe a slight discrepancy between SENTRY results with our runs and numbers originally reported in the paper (Prabhu et al., 2021). We find that this is due to differences in batch size used in original work versus in our runs (which we kept the same for all the algorithms). In App. **??**, we report SENTRY results with the updated batch size. With the new batch size, we reconcile SENTRY results but also observe a significant improvement in FixMatch results. We refer reader to App. **??** for a more detailed discussion.

F.4 Dataset Details

In this section, we provide additional details about the datasets used in our benchmark study.

Dataset	Source	Target
CIFAR10	CIFAR10v1	CIFAR10v1, CIFAR10v2, CIFAR10C-Frost (severity 4), CIFAR10C-Pixelate (severity 5), CIFAR10-C Saturate (severity 5)
CIFAR100	CIFAR100	CIFAR100, CIFAR100C-Fog (severity 4), CIFAR100C-Motion Blur (severity 2), CIFAR100C-Contrast (severity 4), CIFAR100C-spatter (severity 2)
Camelyon	Camelyon (Hospital 1–3)	Camelyon (Hospital 1–3), Camelyon (Hospital 4), Camelyon (Hospital 5)
FMoW	FMoW (2002–’13)	FMoW (2002–’13), FMoW (2013–’16), FMoW (2016–’18)
Entity13	Entity13 (ImageNetv1 sub-population 1)	Entity13 (ImageNetv1 sub-population 1), Entity13 (ImageNetv1 sub-population 2), Entity13 (ImageNetv2 sub-population 1), Entity13 (ImageNetv2 sub-population 2)
Entity30	Entity30 (ImageNetv1 sub-population 1)	Entity30 (ImageNetv1 sub-population 1), Entity30 (ImageNetv1 sub-population 2), Entity30 (ImageNetv2 sub-population 1), Entity30 (ImageNetv2 sub-population 2)
Living17	Living17 (ImageNetv1 sub-population 1)	Living17 (ImageNetv1 sub-population 1), Living17 (ImageNetv1 sub-population 2), Living17 (ImageNetv2 sub-population 1), Living17 (ImageNetv2 sub-population 2)
Nonliving26	Nonliving26 (ImageNetv1 sub-population 1)	Nonliving26 (ImageNetv1 sub-population 1), Nonliving26 (ImageNetv1 sub-population 2), Nonliving26 (ImageNetv2 sub-population 1), Nonliving26 (ImageNetv2 sub-population 2)
Officehome	Product	Product, Art, ClipArt, Real
DomainNet	Real	Real, Painiting, Sketch, ClipArt
Visda	Synthetic (originally referred to as train)	Synthetic, Real-1 (originally referred to as val), Real-2 (originally referred to as test)
Civilcomments	Train	Train, Val and Test (all formed by disjoint partitions of online articles)
Mimic Readmissions	Mimic Readmissions (year: 2008)	Mimic Readmissions (year: 2008), Mimic Readmissions (year: 2009), Mimic Readmissions (year: 2010), Mimic Readmissions (year: 2011), Mimic Readmissions (year: 2012), Mimic Readmissions (year: 2013)
Retiring Adults	Retiring Adults (year: 2014 states: [’MD’, ’NJ’, ’MA’])	Retiring Adults (year: 2015; states: [’MD’, ’NJ’, ’MA’]), Retiring Adults (year: 2016; states: [’MD’, ’NJ’, ’MA’]), Retiring Adults (year: 2017; states: [’MD’, ’NJ’, ’MA’]), Retiring Adults (year: 2018; states: [’MD’, ’NJ’, ’MA’])

Table F.1: Details of the datasets considered in our RLSBench.

- **CIFAR10** We use the original CIFAR10 dataset (Krizhevsky and Hinton, 2009) as the source dataset. For target domains, we consider (i) synthetic shifts (CIFAR10-C) due to common corruptions (Hendrycks and Dietterich, 2019); and (ii) natural distribution shift, i.e., CIFAR10v2 (Recht et al., 2018; Torralba et al., 2008) due to differences in data collection strategy. We randomly sample 3 set of CIFAR-10-C datasets. Overall, we obtain 5 datasets (i.e., CIFAR10v1, CIFAR10v2, CIFAR10C-Frost (severity 4), CIFAR10C-Pixelate (severity 5), CIFAR10-C Saturate (severity 5)).

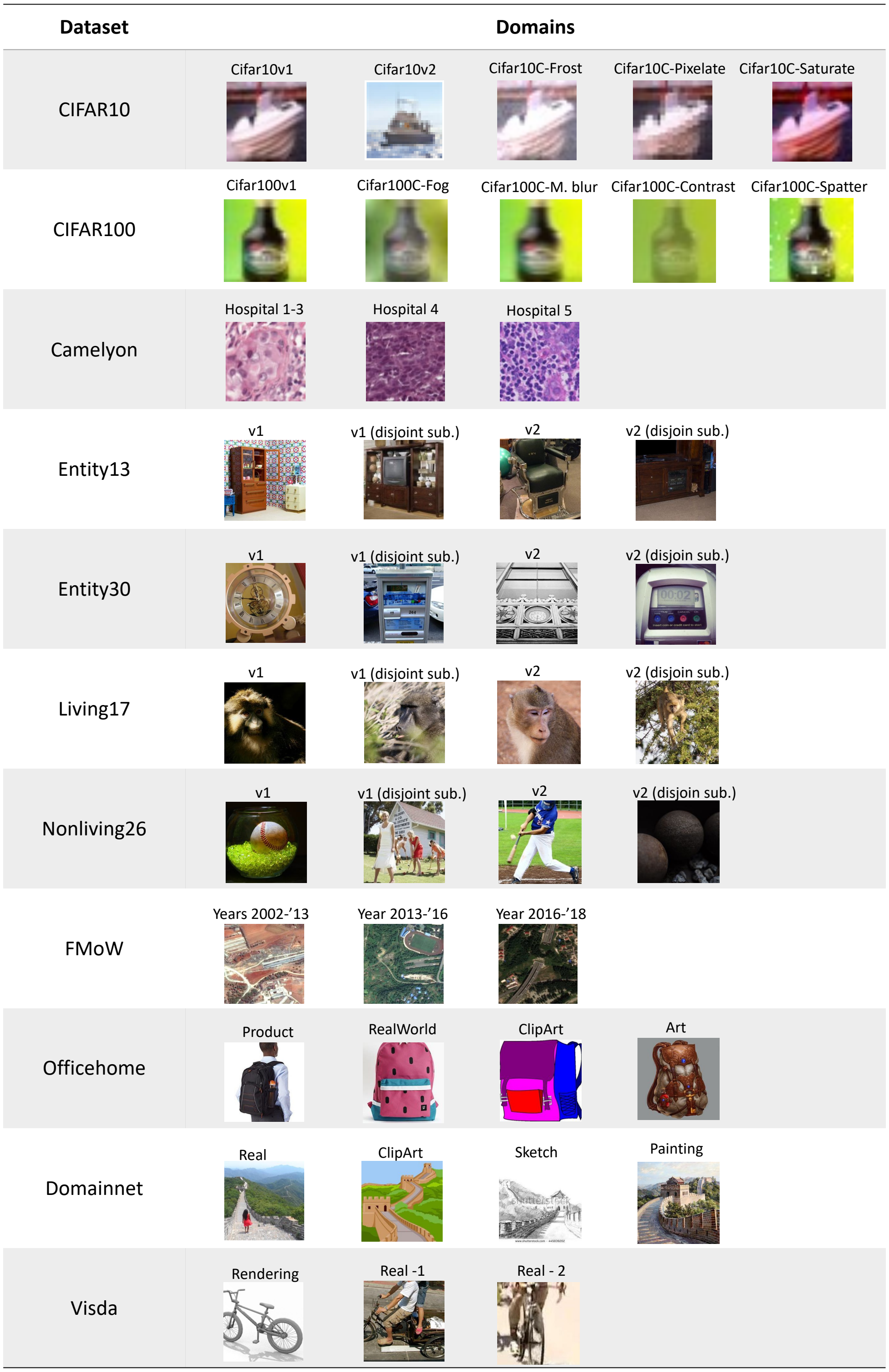


Figure F.4: Examples from all the domains in each vision dataset.

- **CIFAR100** Similar to CIFAR10, we use the original CIFAR100 set as the source dataset. For target domains we consider synthetic shifts (CIFAR100-C) due to common corruptions. We sample 4 CIFAR100-C datasets, overall obtaining 5 domains (i.e., CIFAR100, CIFAR100C-Fog (severity 4), CIFAR100C-Motion Blur (severity 2), CIFAR100C-Contrast (severity 4), CIFAR100C-spatter (severity 2)).

- **FMoW** In order to consider distribution shifts faced in the wild, we consider FMoW-WILDs (Christie et al., 2018; Koh et al., 2021) from WILDS benchmark, which contains satellite images taken in different geographical regions and at different times. We use the original train as source and OOD val and OOD test splits as target domains as they are collected over different time-period. Overall, we obtain 3 different domains.

- **Camelyon17** Similar to FMoW, we consider tumor identification dataset from the wilds benchmark (Bandi et al., 2018). We use the default train as source and OOD val and OOD test splits as target domains as they are collected across different hospitals. Overall, we obtain 3 different domains.

- **BREEDs** We also consider BREEDs benchmark (Santurkar et al., 2021) in our setup to assess robustness to subpopulation shifts. BREEDs leverage class hierarchy in ImageNet to re-purpose original classes to be the subpopulations and defines a classification task on superclasses. We consider distribution shift due to subpopulation shift which is induced by directly making the subpopulations present in the training and test distributions disjoint. BREEDs benchmark contains 4 datasets **Entity-13**, **Entity-30**, **Living-17**, and **Non-living-26**, each focusing on different subtrees and levels in the hierarchy. We also consider natural shifts due to differences in the data collection process of ImageNet (Russakovsky et al., 2015), e.g, ImageNetv2 (Recht et al., 2019b) and a combination of both. Overall, for each of the 4 BREEDs datasets (i.e., Entity-13, Entity-30, Living-17, and Non-living-26), we obtain four different domains. We refer to them as follows: BREEDsv1 sub-population 1 (sampled from ImageNetv1), BREEDsv1 sub-population 2 (sampled from ImageNetv1), BREEDsv2 sub-population 1 (sampled from ImageNetv2), BREEDsv2 sub-population 2 (sampled from ImageNetv2). For each BREEDs dataset, we use BREEDsv1 sub-population A as source and the other three as target domains.

- **OfficeHome** We use four domains (art, clipart, product and real) from OfficeHome dataset (Venkateswara et al., 2017). We use the product domain as source and the other domains as target.

- **DomainNet** We use four domains (clipart, painting, real, sketch) from the Domainnet dataset (Peng et al., 2019). We use real domain as the source and the other domains as target.

- **Visda** We use three domains (train, val and test) from the Visda dataset (Peng et al., 2018). While 'train' domain contains synthetic renditions of the objects, 'val' and 'test' domains contain real world images. To avoid confusing, the domain names with their roles as splits, we rename them as 'synthetic', 'Real-1' and 'Real-2'. We use the synthetic (original train set) as the source domain and use the other domains as target.

- **Civilcomments** (Borkan et al., 2019) from the wilds benchmark which includes three domains: train, OOD val, and OOD test, for toxicity detection with domains corresponding to different demographic subpopulations. The dataset has subpopulation shift across different demographic groups as the dataset in each domain is collected from a different partition of online articles.

- **Retiring Adults** (Ding et al., 2021) where we consider the ACSIncome prediction task with various domains representing different states and time-period; We randomly select three states and consider dataset due to shifting time across those states. Details about precise time-periods and states are in Table F.1.

- **Mimic Readmission** (Johnson et al., 2020; PhysioBank, 2000) where the task is to predict readmission risk with various domains representing data from different time-period. Details about precise time-periods are in Table F.1.

We provide scripts to setup these datasets with single command in our code. To investigate the performance of different methods under the stricter label shift setting, we also include a hold-out partition of source domain in the set of target domains. For these distribution shift pairs where source and target domains are i.i.d. partitions, we obtain the stricter label shift problem. We summarize the information about source and target domains in Table F.1.

Train-test splits We partition each source and target dataset into 80% and 20% i.i.d. splits. We use 80% splits for training and 20% splits for evaluation (or validation). We throw away labels for the 80% target split and only use labels in the 20% target split for final evaluation. The rationale behind splitting the target data is to use a completely unseen batch of data for evaluation. This avoids evaluating on examples where a model potentially could have overfit. over-fitting to unlabeled examples for evaluation. In practice, if the aim is to make predictions on all the target data (i.e., transduction), we can simply use the (full) target set for training and evaluation.

F.5 Illustration of Our Proposed Meta=algorithm

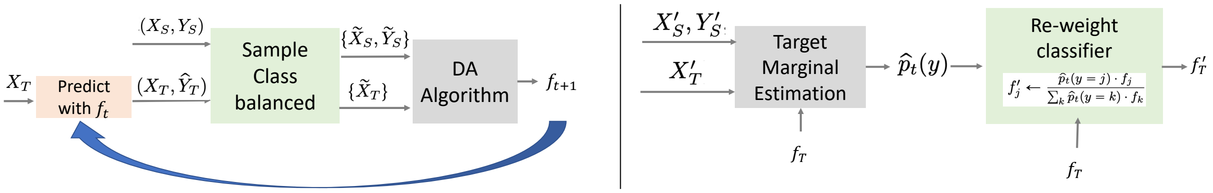


Figure F.5: **(left)** Illustration of RS method at every iteration. **(right)** Illustration of post-hoc reweighting of the classifier with RW method.

F.6 Methods to estimate target marginal under the stricter label shift assumption

In this section, we describe the methods proposed to estimate the target label marginal under the stricter label shift assumption. Recall that under the label shift assumption, $p_s(y)$ can differ from $p_t(y)$ but the class conditional stays the same, i.e., $p_t(x|y) = p_s(x|y)$. We focus our discussion on recent methods that leverage off-the-shelf classifier to yield consistent estimates under mild assumptions (Alexandari et al., 2021; Azizzadenesheli et al., 2019; Garg et al., 2020a; Lipton et al., 2018b). For simplicity, we assume we possess labeled source data $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ and unlabeled target data $\{x_{n+1}, x_{n+2}, \ldots, x_{n+m}\}$.

RLLS First, we discuss *Regularized Learning under Label Shift* (RLLS) (Azizzadenesheli et al., 2019) (a variant of *Black Box Shift Estimation* (BBSE, Lipton et al. (2018b))): moment-matching based estimators that leverage (possibly biased, uncalibrated, or inaccurate) predictions to estimate the shift. RLLS solves the following optimization problem to estimate the importance weights $w_t(y) = \frac{p_t(y)}{p_s(y)}$ as:

$$\widehat{w}w_t^{\text{RLLS}} = \operatorname*{arg\,min}_{w \in \mathcal{W}} \left\| \widehat{w}C_f w - \widehat{w}\mu_f \right\| 2 + \lambda_{\text{RLLS}} \left\| w - 1 \right\| 2 \,. \tag{F.1}$$

where $\mathcal{W} = \{w \in \mathbb{R}^d | \sum_y w(y)p_s(y) = 1 \text{ and } \forall y \in \mathcal{Y} \quad w(y) > 0\}$. $\widehat{w}C_f$ is empirical confusion matrix of the classifier f on source data and $\widetilde{\mu}_f$ is the empirical average of predictions of the classifier f on unlabeled target data. With labeled source data data, the empirical confusion matrix can be computed as:

$$[\widehat{w}C_f]_{i,j} = \frac{1}{n} \sum_{k=1}^{n} f_i(x_k) \cdot \mathbb{I}\left[y_k = j\right] \,.$$

To estimate target label marginal, we can multiple the estimated importance weights with the source label marginal (we can estimate source label marginal simply from labeled source data).

In our relaxed label shift problem, we use validation source data to compute the confusion matrix and use hold portion of target unlabeled data to compute μ_f. Unless specified otherwise, we use RLLS to estimate the target label marginal throughout the paper. We choose λ_{RLLS} as suggested in the original paper (Azizzadenesheli et al., 2019).

MLLS Next, we discuss Maximum Likelihood Label Shift (MLLS) (Alexandari et al., 2021; Saerens et al., 2002): an Expectation Maximization (EM) algorithm that maximize the likelihood of observed unlabeled target data to estimate target label marginal assuming access to a classifier that outputs the source calibrated probabilities. In particular, MLLS uses the following objective:

$$\widehat{w}w_t^{\text{MLLS}} = \operatorname*{arg\,min}_{w \in \mathcal{W}} \frac{1}{m} \sum_{i=1} \log(w^T f(x_{i+n})) \,, \tag{F.2}$$

where f is the classifier trained on source and $\mathcal{W}$ is the same constrained set defined above. We can again estimate the target label marginal by simply multiplying the estimated importance weights with the source label marginal.

Baseline estimator Given a classifier f, we can estimate the target label marginal as simply the average of the classifier output on unlabeled target data, i.e.,

$$\widehat{w}p_t^{\text{baseline}} = \frac{1}{m}\sum_{i=1} f(x_{i+n}) \,. \tag{F.3}$$

Note that all of the methods discussed before leverage an off-the-shelf classifier f. Hence, we experiment with classifiers obtained with various deep domain adaptation heuristics to estimate the target label marginal.

Having obtained an estimate of target label marginal, we can simply re-weight the classifier with $\widehat{w}p_t$ as $f'_j = \frac{\widehat{w}p_t(y=j)\cdot f_j}{\sum_k \widehat{w}p_t(y=k)\cdot f_k}$ for all $j \in \mathcal{Y}$. Note that, if we train f on a non-uniform source class-balance (and without re-balancing as in Step 1 of Algorithm 9), then we can re-weight the classifier with importance-weights $\widehat{w}w_t$ as $f'_j = \frac{\widehat{w}w_t(y=j)\cdot f_j}{\sum_k \widehat{w}w_t(y=k)\cdot f_k}$ for all $j \in \mathcal{Y}$.

F.7 Theoretical Definition for Relaxed Label Shift

Domain adaptation problems are, in general, ill-posed (Ben-David et al., 2010c). Several attempts have been made to investigate additional assumptions that render the problem well-posed. One such example includes the label-shift setting, where $p(x|y)$ does not change but that $p(y)$ can. Under label shift, two challenges arise: (i) estimate the target label marginal $p_t(y)$; and (ii) train a classifier f to maximize the performance on the target domain. However, these assumptions are typically, to some degree, violated in practice. This paper aims to relax this assumption and focuses on *relaxed label shift* setting. In particular, we assume that the label distribution can shift from source to target arbitrarily but that $p(x|y)$ varies between source and target in some comparatively restrictive way (e.g., shifts arising naturally in the real world like ImageNet (Russakovsky et al., 2015) to ImageNetV2 (Recht et al., 2019b)).

Mathematically, we assume a divergence-based restriction on $p(x|y)$, i.e., for some small $\epsilon > 0$ and distributional distance $\mathcal{D}$, we have $\max_y \mathcal{D}(p_t(x|y), p_t(x|y)) \leqslant \epsilon$ but allowing an arbitrary shift in the label marginal $p(y)$. Previous works have defined these constraints in different ways (Kumar et al., 2020; Tachet des Combes et al., 2020; Wu et al., 2019).

In particular, we can use Wasserstein-infinity distance to define our constraint. First, we define Wasserstein given probability measures p, q on $\mathcal{X}$:

$$W_\infty(p,q) = \inf\{\sup_{x\in\mathbb{R}^d} \|f(x)-x\|\,2 : f:\mathbb{R}^d \to \mathbb{R}^d, f_{\#}p = q\},$$

where $\#$ denotes the push forward of a measure, i.e., for every set $S \subseteq \mathbb{R}^d, p(S) = p(f^{-1}(S))$. Intuitively, W_∞ moves points from the distribution p to q by distance at most ϵ to match the distributions. Hence, our $D := \max_y W_\infty(p_s(x|y), p_t(x|y)) \leqslant \epsilon$. Similarly, we can define our distribution constraint in KL or TV distances. We can define our constraint in a representation space $\mathcal{Z}$ obtained by projection inputs $x \in \mathcal{X}$ with a function $h : \mathcal{X} \to \mathcal{Z}$. Intuitively, we want to define the distribution distance with some h that captures all the required information for predicting the label of interest but satisfies a small distributional divergence in the projected space. However, in practice, it's hard to empirically verify these distribution distances for small enough ϵ with finite samples. Moreover, we lack a rigorous characterization of the sense in which those shifts arise in popular DA benchmarks, and since, the focus of our work is on the empirical evaluation with real-world datasets, we leave a formal investigation for future work. .

F.8 Target Marginal Estimation and its Effect on Accuracy

F.8.1 Tabular Datasets

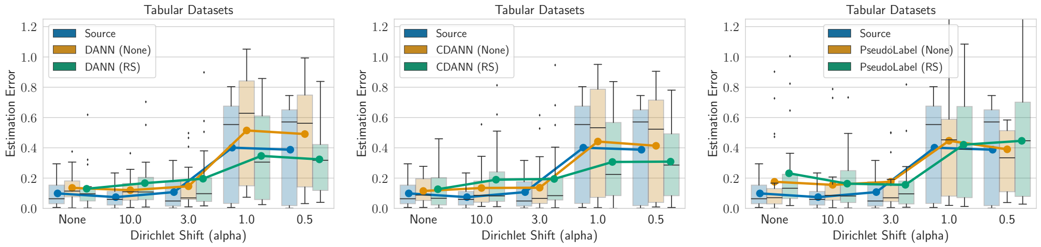


Figure F.6: Target label marginal estimation (ℓ_1) error with RLLS and classifiers obtained with different DA methods

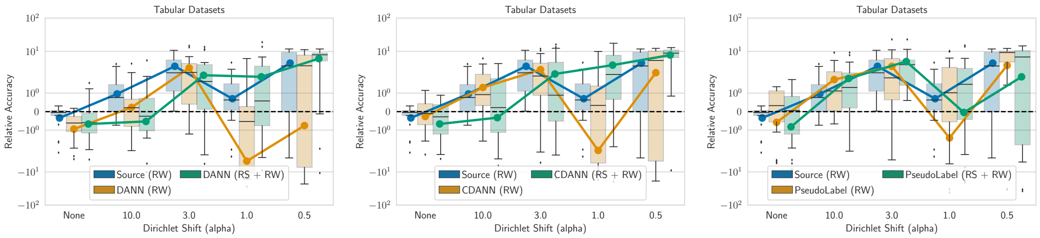


Figure F.7: Relative performance of DA methods when paired with RW corrections

Figure F.8: *Target label marginal estimation (ℓ_1) error and relative performance with RLLS and classifiers obtained with different DA methods.* For tabular datasets, RLLS with classifiers obtained with DA methods improves over RLLS with a source-only classifier for severe target label marginal shifts. Correspondingly for severe target label marginal shifts, we see improved performance with post-hoc RW correction applied to classifiers trained with DA methods as compared to when applied to source-only models.

F.8.2 Vision Datasets

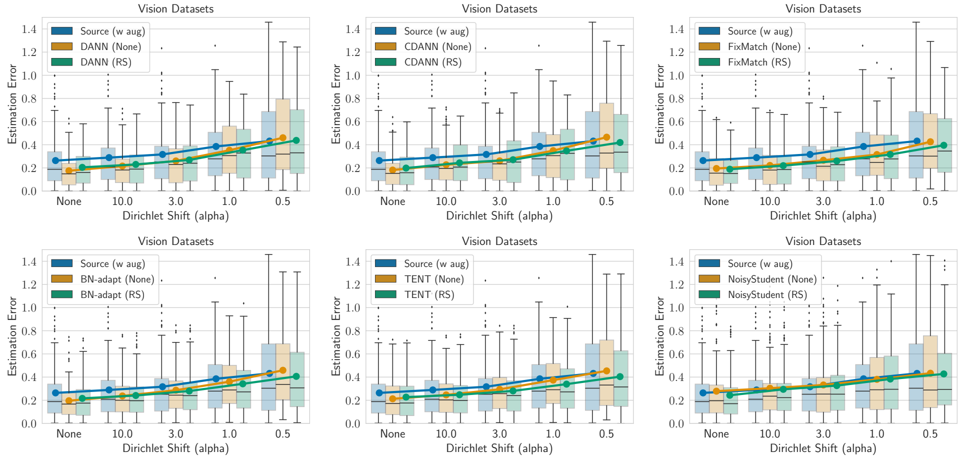


Figure F.9: Target label marginal estimation (ℓ_1) error with RLLS and classifiers obtained with different DA methods

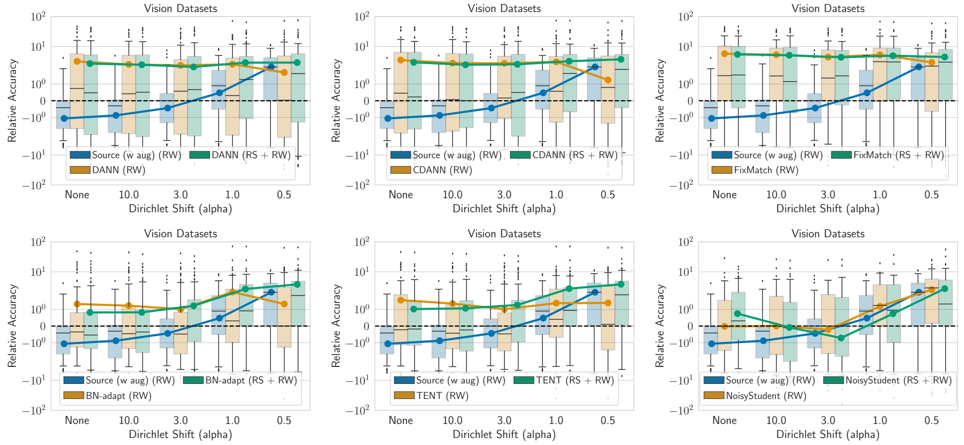


Figure F.10: Relative performance of DA methods when paired with RW corrections

Figure F.11: *Target label marginal estimation (ℓ_1) error and relative performance with RLLS and classifiers obtained with different DA methods.* Across all shift severities (except for $\alpha = 0.5$) in vision datasets, RLLS with classifiers obtained with DA methods improves over RLLS with a source-only classifier. Correspondingly, we see significantly improved performance with post-hoc RW correction applied to classifiers trained with DA methods as compared to when applied to source-only models.

F.8.3 NLP Datasets

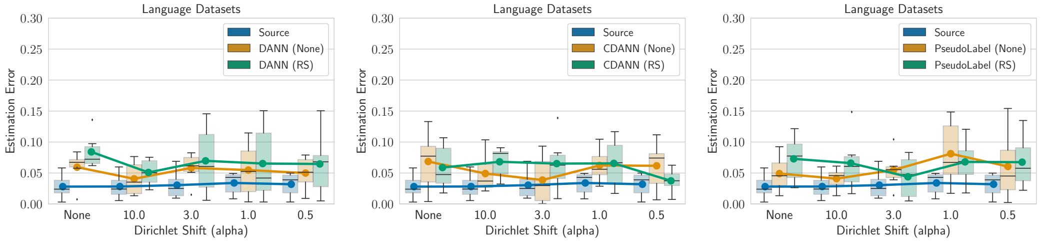


Figure F.12: Target label marginal estimation (ℓ_1) error with RLLS and classifiers obtained with different DA methods

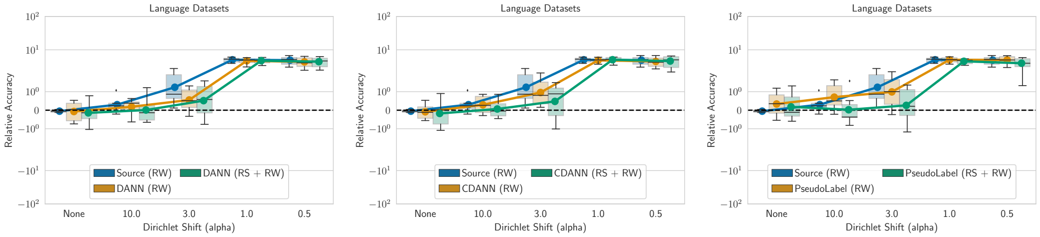


Figure F.13: Relative performance of DA methods when paired with RW corrections

Figure F.14: *Target label marginal estimation (ℓ_1) error and relative performance with RLLS and classifiers obtained with different DA methods.* For NLP datasets, RLLS with source-only classifiers performs better than RLLS with classifiers obtained with DA methods. Correspondingly, we see improved performance with post-hoc RW correction applied to source-only models over classifiers trained with DA methods.

F.8.4 Comparison of different target label marginal estimation methods

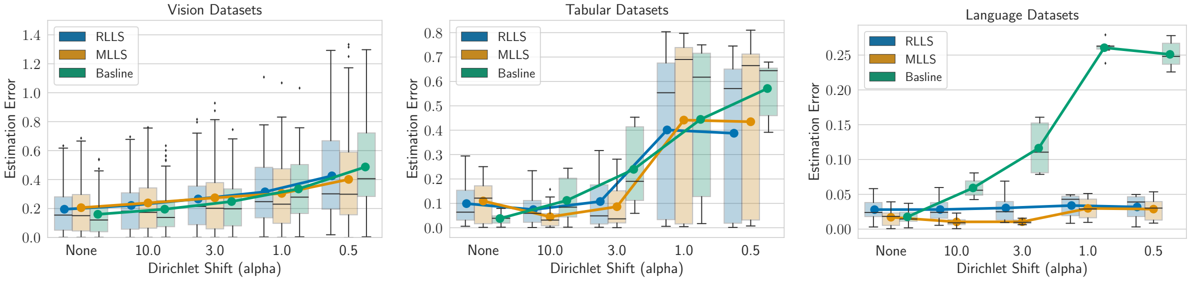


Figure F.15: *Comparison of different target label marginal estimation methods.* We plot estimation errors with different methods with the source-only classifier. For all modalities, we observe a trade-off between estimation error with the baseline method and RLLS (or MLLS) method with severity in target marginal shift.

F.9 Results with Oracle Early Stopping Criterion

In this section, we report results with oracle early stopping criterion. On vision and tabular datasets, we observe differences in performance when using target performance versus source hold-out performance for model selection. This highlights a more nuanced behavior than the accuracy-on-the-line phenomena (Miller et al., 2021; Recht et al., 2019b). We hope to study this contrasting behavior in more detail in future work.

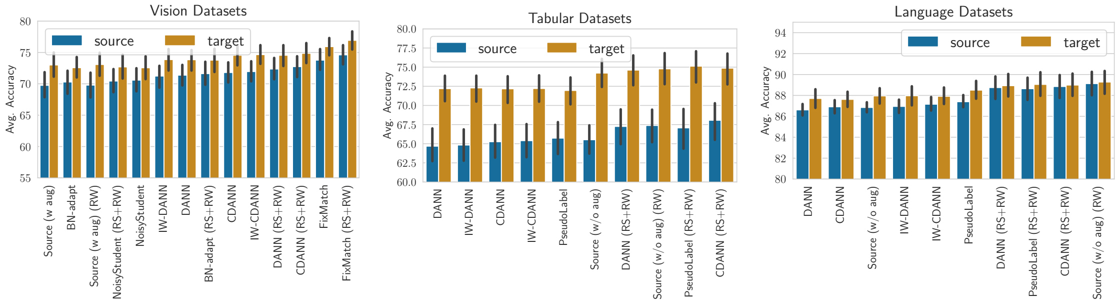


Figure F.16: *Average accuracy of different DA methods aggregated across all distribution pairs in each modality.* We compare the performance with early stopping point obtained with source validation performance and target validation performance.

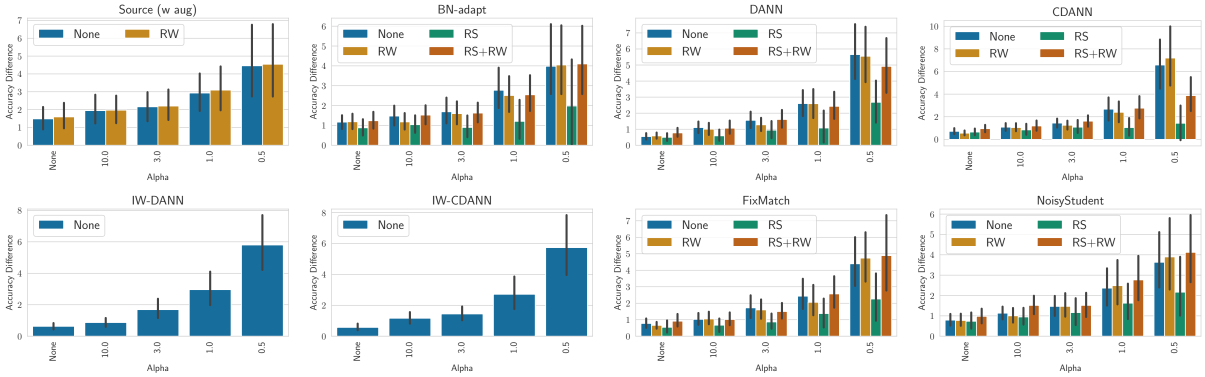


Figure F.17: *Accuracy difference between using source and target performance as early stopping criteria for different DA methods aggregated across all distribution shift pairs in vision datasets.* We observe that as the shift severity increases (i.e., as α decreases), the accuracy difference increases for all the methods.

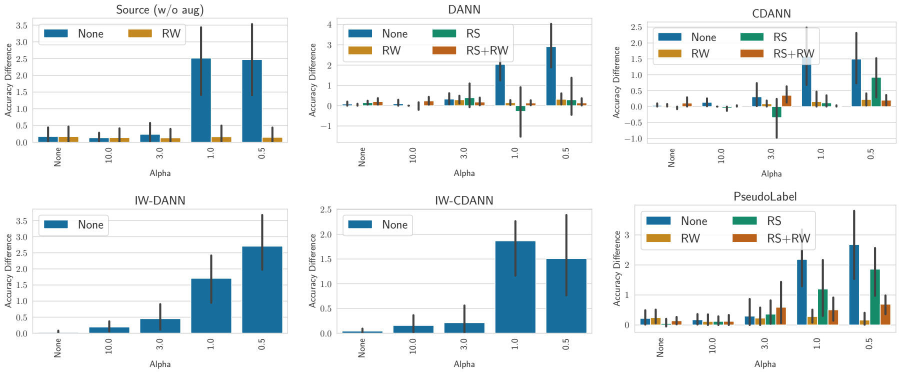


Figure F.18: *Accuracy difference between using source and target performance as early stopping criteria for different DA methods aggregated across all distribution shift pairs in language datasets.* We observe that as the shift severity increases (i.e., as α decreases), the accuracy difference increases for all the methods without any correction. With RS and RW corrections, we observe that the accuracy difference remains relatively constant as the shift severity increases.

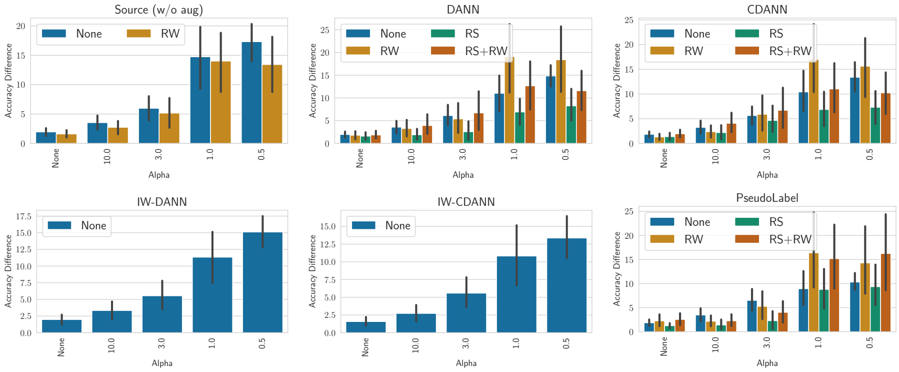


Figure F.19: *Accuracy difference between using source and target performance as early stopping criteria for different DA methods aggregated across all distribution shift pairs in tabular datasets.* We observe that as the shift severity increases (i.e., as α decreases), the accuracy difference increases for all the methods.

F.10 Aggregate Accuracy with Different DA methods on Each Dataset

Dataset	Source	DANN	IW-DANN	CDANN	IW-CDANN	PseudoLabel
Civilcomments	86.85	86.62	86.95	86.91	87.16	87.4

Dataset	Source		DANN				CDANN				PseudoLabel			
	None	RW	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW
Civilcomments	86.8	89.1	86.6	88.8	87.1	88.8	86.9	89.0	86.9	88.9	87.4	89.3	86.9	88.6

Table F.2: *Results with different DA methods on NLP datasets aggregated across target label marginal shifts.*

Dataset	Source	DANN	IW-DANN	CDANN	IW-CDANN	PseudoLabel
Retiring Adult	77.44	77.17	77.35	78.15	78.44	78.30
Mimic Readmission	57.57	56.36	56.48	56.67	56.71	57.35

Dataset	Source		DANN				CDANN				PseudoLabel			
	None	RW	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW
Retiring Adults	77.4	80.0	77.2	79.5	77.4	79.4	78.1	80.5	78.1	80.4	78.3	80.8	78.5	80.8
Mimic Readmissions	57.6	59.0	56.4	55.1	57.3	59.2	56.7	56.8	57.4	59.9	57.4	57.7	57.7	57.9

Table F.3: *Results with different DA methods on tabular datasets aggregated across target label marginal shifts.*

Dataset	Source (wo aug)	Source (w aug)	BN-adapt	TENT	DANN	IW-DAN	CDAN	IW-CDAN	Fix-Match	Noisy-Student	Sentry
CIFAR-10	89.69	89.14	89.21	89.20	90.86	90.78	90.00	89.93	91.87	90.72	91.83
CIFAR-100	65.99	76.69	77.57	77.58	74.80	74.81	74.57	74.66	79.03	77.60	74.74
FMoW	64.00	68.99	65.52	66.55	60.11	60.33	60.79	61.05	68.37	68.90	51.06
Camelyon	77.42	76.95	85.70	82.48	86.66	85.89	85.45	84.27	86.29	79.29	86.81
Domainnet	52.37	50.50	50.66	51.12	51.91	52.05	54.40	54.29	57.96	51.49	55.16
Entity13	76.93	80.07	77.99	78.04	78.26	78.75	79.74	79.28	80.25	80.37	73.58
Entity30	62.61	69.83	68.09	68.09	67.90	68.36	68.51	69.34	69.95	69.10	58.51
Living17	64.13	69.30	68.84	68.82	72.12	69.87	70.72	70.65	72.86	72.16	53.44
Nonliving26	54.75	63.95	62.60	63.02	61.69	61.99	62.53	64.51	62.98	63.60	44.82
Officehome	59.89	59.45	60.59	60.82	66.05	65.79	66.19	66.15	65.48	60.47	65.37
Visda	58.47	53.41	59.98	60.96	69.69	69.79	72.55	72.80	72.02	53.51	72.23
Avg	66.02	68.94	69.70	69.70	70.92	70.77	71.40	71.54	73.37	69.75	66.14

Table F.4: *Results with different DA methods on vision datasets aggregated across target label marginal shifts.* While no single DA method performs consistently across different datasets, FixMatch seems to provide the highest aggregate improvement over a source-only classifier in our testbed.

Dataset	**Source**		**BN-adapt**				**CDANN**				**FixMatch**			
	None	RW	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW
CIFAR-10	89.1	89.4	89.2	91.4	92.1	92.9	90.0	91.3	91.4	92.5	91.9	93.1	93.6	94.1
CIFAR-100	76.7	77.5	77.6	78.8	77.9	79.0	74.6	75.8	74.1	75.3	79.0	79.6	79.1	79.8
FMoW	69.0	70.3	65.5	67.2	66.2	65.6	60.8	61.9	57.0	55.2	68.4	69.4	64.9	66.7
Camelyon	77.0	77.9	85.7	85.9	88.5	89.3	85.5	85.8	87.9	88.5	86.3	87.0	86.6	86.8
Domainnet	50.5	48.2	50.7	50.1	51.4	49.8	54.4	54.2	54.7	54.3	58.0	57.5	58.4	57.8
Entity13	80.1	80.9	78.0	79.4	79.8	80.7	79.7	80.2	80.6	81.4	80.3	81.9	81.4	82.4
Entity30	69.8	70.1	68.1	69.2	69.1	70.0	68.5	69.6	69.4	70.5	70.0	71.6	70.1	71.2
Living17	69.3	69.9	68.8	69.7	69.6	70.1	70.7	71.3	72.9	72.7	72.9	72.8	72.3	71.9
Nonliving26	63.9	64.5	62.6	63.0	63.7	63.9	62.5	62.9	63.8	64.0	63.0	64.7	63.9	64.8
Officehome	59.4	57.9	60.6	60.5	60.9	60.4	66.2	66.3	66.1	65.1	65.5	64.9	66.5	66.1
Visda	53.4	52.1	60.0	60.6	59.5	58.8	72.6	72.6	75.3	75.3	72.0	72.5	73.5	73.8
Avg	68.9	69.0	69.7	70.5	70.8	70.9	71.4	72.0	72.1	72.3	73.4	74.1	73.7	74.1

Dataset	**TENT**				**DANN**				**NoisyStudent**			
	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW
CIFAR-10	89.2	91.4	92.1	92.9	90.9	92.3	91.5	92.6	90.7	90.8	90.6	90.7
CIFAR-100	77.6	78.8	78.0	79.0	74.8	75.9	74.8	76.1	77.6	78.0	77.9	78.0
FMoW	66.6	67.4	66.7	66.1	60.1	61.6	56.4	54.5	68.9	69.8	67.1	68.0
Camelyon	82.5	82.7	87.8	88.9	86.7	87.3	88.4	88.8	79.3	79.1	79.2	79.3
Domainnet	51.1	50.6	51.8	50.3	51.9	52.1	53.6	53.5	51.5	49.8	51.3	49.5
Entity13	78.0	79.5	79.8	80.8	78.3	79.4	79.7	80.8	80.4	81.5	80.6	81.7
Entity30	68.1	69.2	69.1	70.1	67.9	69.2	69.0	69.8	69.1	70.1	69.3	70.3
Living17	68.8	69.7	69.6	70.1	72.1	73.0	71.8	72.3	72.2	71.1	69.3	69.4
Nonliving26	63.0	63.4	63.3	63.8	61.7	62.4	63.1	63.0	63.6	64.3	63.2	64.8
Officehome	60.8	60.4	60.9	60.4	66.1	66.1	66.5	65.3	60.5	59.5	60.8	59.5
Visda	61.0	61.5	60.3	59.6	69.7	69.9	73.1	73.2	53.5	51.5	55.7	54.3
Avg	69.7	70.4	70.8	71.1	70.9	71.7	71.6	71.8	69.7	69.6	69.5	69.6

Table F.5: *Results with DA methods paired with re-sampling (RS) and re-weighting (RW) correction (with RLLS estimate) aggregated across target label marginal shifts for vision datasets.* RS and RW seem to help for all datasets and they both together significantly improve aggregate performance over no correction for all DA methods.

F.11 Description of Deep Domain Adaptation Methods

In this section, we summarize deep DA methods compared in our RLSBENCH testbed. We also discuss how each method combines with our meta-algorithm to handle shift in class proportion.

F.11.1 Source only training

We consider empirical risk minimization on the labeled source data as a baseline. Since this simply ignores the unlabeled target data, we call this as source only training. As mentioned in the main paper, we perform source only training with and without data augmentations. Formally, we minimize the following ERM loss:

$$L_{\text{source only}}(f) = \frac{1}{n}\sum_{i=1}^{n} \ell(f(T(x_i), y_i)) \,, \tag{F.4}$$

where T is the stochastic data augmentation operation for vision datasets and ℓ is a loss function. For NLP and tabular datasets, T is the identity function. Throughout the paper, we use cross-entropy loss minimization. Unless specified otherwise, we use strong augmentations as the data augmentation technique for vision datasets. For NLP and tabular datasets, we do not use any data augmentation.

As mentioned in the main paper, we do not include re-sampling results with a source only model as it is trained only on source data and we observed no differences with just balancing the source data (as for most datasets source is already balanced) in our experiments. After obtaining a classifier f, we can first estimate the target label marginal and then adjust the classifier f with post-hoc re-weighting with importance ratios $w_t(y) = \widehat{w}p_t(y)/\widehat{w}p_s(y)$.

Adversarial training of a source only model Along with standard training of a source only model with data augmentation, we experiment with adversarially robust models (Madry et al., 2017). To train adversarially robust models, we replace the standard ERM objective with a robust risk minimization objective:

$$L_{\text{source only (adv)}}(f) = \frac{1}{n}\sum_{i=1}^{n} \ell(R(T(x_i), y_i), y_i) \,, \tag{F.5}$$

where $R(\cdot)$ performs the adversarial augmentation. In our paper, we use targeted Projected Gradient Descent (PGD) attacks with ℓ_2 perturbation model.

F.11.2 Domain-adversarial training methods

Domain-adversarial trianing methods aim to learn domain invariant feature representations. These methods aimed at practical problems with non-overlapping support and are motivated by theoretical results showing that the gap between in- and out-of-distribution performance depends on some measure of divergence between the source and target distributions (Ben-David et al., 2010a; Ganin et al., 2016). While simultaneously minimizing the source error, these methods align the representations between source and target distribution. To perform alignment, these methods penalize divergence between feature representations across domains, encouraging the model to produce feature representations that are similar across domain.

Before describing these methods, we first define some notation. Consider a model $f = g \circ h$, where $h : \mathcal{X} \to \mathbb{R}^d$ is the featurizer that maps the inputs to some d dimensional feature space, and the head $g : \mathbb{R}^d \to \Delta^{k-1}$ maps the features to the prediction space. Following Sagawa et al. (2021), with all of our domain invariant methods, we use strong augmentations with source and target data for vision datasets. For NLP and tabular datasets, we do not use any data augmentation.

DANN DANN was proposed in Ganin et al. (2016). DANN approximates the divergence between feature representations of source and target domain by leveraging a domain discriminator classifier. Domain discriminator f_d aims to discriminate between source and target domains. Given a batch of inputs from source and target, this deep network f_d classifies whether the examples are from the source data or target data. In particular, the following loss function is used:

$$L_{\text{domain disc.}}(f_d) = \frac{1}{n}\sum_{i=1}^{n} \ell(f_d(h(T(x_i))), 0) + \frac{1}{m}\sum_{i=n+1}^{n+m} \ell(f_d(h(T(x_i))), 1)\,, \tag{F.6}$$

where $\{x_1, x_2, \ldots, x_n\}$ are n source examples and $\{x_{n+1}, \ldots, x_{m+n}\}$ are m target examples. Overall, the following loss function is used to optimize models with DANN:

$$L_{\text{DANN}}(h, g, f_d) = L_{\text{source only}}(g \circ h) - \lambda L_{\text{domain disc.}}(f_d)\,. \tag{F.7}$$

$L_{\text{DANN}}(h, g, f_d)$ is maximized with respect to the domain discriminator classifier and $L_{\text{DANN}}(h, g, f_d)$ minimized with respect to the underlying featurize and the source classifier. This is achieved by gradient reversal layer in practice. To train, three networks, we use three different learning rate η_f, η_g, and η_{f_d}. We discuss these hyperparameter details in App. F.12. We adapted our DANN implementation from Sagawa et al. (2021) and Transfer learning library (Jiang et al., 2022).

CDANN Conditional Domain adversarial neural network is a variant of DANN (Long et al., 2018). Here the domain discriminator is conditioned on the classifier g's prediction. In particular, instead of training the domain discriminator on the representation output of h, these methods operate on the outer product between the feature presentation $h(x)$ at an input x and the classifier's probabilistic prediction $f = g \circ h(x)$ (i.e., $h(x) \otimes f(x)$). Thus instead of training the domain discriminator classifier f_d on the d dimensional input space, they train it on $d \times k$ dimensional space. In particular, the following loss function is used:

$$L_{\text{CDAN domain disc.}}(f_d, g, h) = \frac{1}{n}\sum_{i=1}^{n} \ell(f_d(f \otimes h(T(x_i))), 0) + \frac{1}{n}\sum_{i=n+1}^{n+m} \ell(f_d(f \otimes h(T(x_i))), 1)\,, \tag{F.8}$$

where $\{x_1, x_2, \ldots, x_n\}$ are n source examples and $\{x_{n+1}, \ldots, x_{m+n}\}$ are m target examples. The overall loss is the same as DANN where $L_{\text{domain disc.}}(f_d)$ is replaced with $L_{\text{CDAN domain disc.}}(f_d, g, h)$.

We adapted our implementation for CDANN from Transfer learning library (Jiang et al., 2022).

To adapt DANN and CDANN to our meta algorithm, at each epoch we can perform re-balancing of source and target data as in Step 1 and 4 of Algorithm 9. After obtaining the classifier f, we can use this classifier to first obtain an estimate of the target label marginal and then perform re-weighting adjustment with the obtained estimate.

IW-DANN and IW-CDANN Tachet et al. (2020) proposed training with importance re-weighting correction with DANN and CDANN objectives to accommodate for the shift in the target label proportion. In particular, at every epoch of training they first estimate the importance ratio $\widehat{w}w_t$ (with BBSE on training source and training target data) and then re-weight the domain discriminator objective and ERM objective. In particular, the domain discriminator loss for IW-DANN can be written as:

$$L^{\widehat{w}w}_{\text{domain disc.}}(f_d) = \frac{1}{n}\sum_{i=1}^{n} \widehat{w}w(y_i)\ell(f_d(h(T(x_i))), 0) + \frac{1}{n}\sum_{i=n+1}^{n+m} \ell(f_d(h(T(x_i))), 1)\,, \tag{F.9}$$

where we multiply the source loss with importance weights. Similarly, we can re-write the source only training objective with importance re-weighting as follows:

$$L^{\widehat{w}w}_{\text{source only}}(f) = \frac{1}{n}\sum_{i=1}^{n} \widehat{w}w(y_i)\ell(f(T(x_i), y_i))\,. \tag{F.10}$$

Overall, the following objective is used to optimize models with IW-DANN:

$$L_{\text{IW-DANN}}(h, g, f_d) = L^{\widehat{w}w}_{\text{source only}}(g \circ h) - \lambda L^{\widehat{w}w}_{\text{domain disc.}}(f_d)\,, \tag{F.11}$$

where the importance weights are updated after every epoch with classifier obtained in previous step. Similarly, with using importance re-weights with the CDANN objective, we obtain IW-CDANN objective.

In population, IW-CDANN and IW-DANN correction matches the correction with our meta-algorithm for DANN and CDANN. However, the behavior this importance re-weighting correction can be different from our meta-algorithm for over-parameterized models with finite data (Byrd and Lipton, 2019). Recent empirical and theoretical findings have highlighted that importance re-weighting have minor to no effect on overparameterized models when trained for several epochs (Byrd and Lipton, 2019; Xu et al., 2021). On the other hand, with finite samples, re-sampling (when class labels are available) has shown different and promising empirical behavior (An et al., 2020; Idrissi et al., 2022). This may highlight the differences in the behavior of IW-CDANN (or IW-DANN) with our meta algorithm on CDANN (or DANN).

We refer to the implementation provided by the authors (Tachet et al., 2020).

F.11.3 Self-training methods

Self-training methods leverage unlabeled data by 'pseudo-labeling' unlabeled examples with the classifier's own predictions and training on them as if they were labeled examples. Recent self-training methods also often make use of consistency regularization, for example, encouraging the model to make similar predictions on augmented versions of unlabeled example. In our work, we experiment with the following methods:

PseudoLabel (Lee et al., 2013) proposed PseudoLabel that leverages unlabeled examples with classifier's own prediction. This algorithm dynamically generates psuedolabels and overfits on them in each batch. In particular, while pseudolabels are generated on unlabeled examples, the loss is computed with respect to the same label. PseudoLabel only overfits to the assigned label if the confidence of the prediction is greater than some threshold τ.

Refer to T as the data-augmentation technique (i.e., identity for NLP and tabular datasets and strong augmentation for vision datasets). Then, PseudoLabel uses the following loss function:

$$L_{\text{PseudoLabel}}(f) = \frac{1}{n}\sum_{i=1}^{n} \ell(f(T(x_i), y_i)) + \frac{\lambda_t}{m}\sum_{i=n+1}^{m+n} \ell(f(T(x_i), \widetilde{y}_i)) \cdot \mathbb{I}\left[\max_y f_y(T(x_i)) \geqslant \tau\right],$$

where $\widetilde{y}_i = \arg\max_y f_y(T(x_i))$. PseudoLabel increases λ_t between labeled and unlabeled losses over epochs, initially placing 0 weight on unlabeled loss and then linearly increasing the unlabeled loss weight until it reaches the full value of hyperparameter λ at some threshold step. We fix the step at which λ_t reaches its maximum value λ be 40% of the total number of training steps, matching the implementation to (Sagawa et al., 2021; Sohn et al., 2020).

FixMatch Sohn et al. (2020) proposed FixMatch as a variant of the simpler Pseudo-label method (Lee et al., 2013). This algorithm dynamically generates psuedolabels and overfits on them in each batch. FixMatch employs consistency regularization on the unlabeled data. In particular, while pseudolabels are generated on a weakly augmented view of the unlabeled examples, the loss is computed with respect to predictions on a strongly augmented view. The intuition behind such an update is to encourage a model to make predictions on weakly augmented data consistent with the strongly augmented example. Moreover, FixMatch only overfits to the assigned labeled with weak augmentation if the confidence of the prediction with strong augmentation is greater than some threshold τ.

Refer to T_{weak} as the weak-augmentation and T_{strong} as the strong-augmentation function. Then, FixMatch uses the following loss function:

$$\begin{aligned} L_{\text{FixMatch}}(f) = & \frac{1}{n}\sum_{i=1}^{n} \ell(f(T_{\text{strong}}(x_i), y_i)) \\ & + \frac{\lambda}{m}\sum_{i=n+1}^{m+n} \ell(f(T_{\text{strong}}(x_i), \widetilde{y}_i)) \cdot \mathbb{I}\left[\max_y f_y(T_{\text{strong}}(x_i)) \geqslant \tau\right], \end{aligned}$$

where $\widetilde{y}_i = \arg\max_y f_y(T_{\text{weak}}(x_i))$. We adapted our implementation from Sagawa et al. (2021) which matches the implementation of Sohn et al. (2020) except for one detail. While Sohn et al. (2020) augments labeled examples with weak augmentation, Sagawa et al. (2021) proposed to strongly augment the labeled source examples.

NoisyStudent Xie et al. (2020b) proposed a different variant of Pseudo-labeling. Noisy Student generates pseudolabels, fixes them, and then trains the model (from scratch) until convergence before generating new pseudolabels. Contrast it with FixMatch and PseudoLabel which dynamically generate pseudolabels. The first set of pseudolabels are obtained by training an initial teacher model only on the source labeled data. Then in each iteration, randomly initialized models fit the labeled source data and pseudolabeled target data with pseudolabels assigned by the converged model in the previous iteration. Noisy student objective can be summarized as:

$$L_{\text{NoisyStudent}}(f^N) = \frac{1}{n}\sum_{i=1}^{n} \ell(f^N(T_{\text{strong}}(x_i), y_i)) + \frac{1}{m}\sum_{i=n+1}^{m+n} \ell(f^N(T_{\text{strong}}(x_i), \widetilde{y}_i)),$$

where $\widetilde{y}_i = \arg\max_y f_y^{N-1}(T_{\text{weak}}(x_i))$ is computed with the classifier obtained at $N-1$ step. Note that the randomly initialized model at each iteration uses a dropout of $p = 0.5$ in the penultimate layer. We adopted our implementation of NoisyStudent to Sagawa et al. (2021). To initialize the initial teacher model, we use the source-only model trained with strong augmentations without dropout.

SENTRY Prabhu et al. (2021) proposed a different variant of pseudolabeling method. This method is aimed to tackle DA under relaxed label shift scenario. a SENTRY incorporates a target instance based on its predictive consistency under a committee of strong image transformations. In particular, SENTRY makes N strong augmentations of an unlabeled target example and makes a prediction on those. If the majority of the committee matches the prediction on the sample example with weak-augmentation then entropy is minimized on that example, otherwise the entropy is maximized. Moreover, the authors employ an 'information-entropy' objective aimed to match the prediction at every example with the estimated target label marginal. Overall the SENTRY objective is defined as follows:

$$L_{\text{SENTRY}}(f) = \frac{1}{n}\sum_{i=1}^{n} \ell(f(T_{\text{strong}}(x_i), y_i)) + \frac{1}{m}\sum_{i=n+1}^{m+n}\sum_{j=1}^{k} f_k(y=j|x_i)\log(\widetilde{p}_t(y=j))$$
$$+ \lambda_{\text{unsup}}\frac{1}{m}\sum_{i=n+1}^{m+n}\sum_{j=1}^{k} -f_k(y=j|x_i)\log(f_k(y=j|x_i))\cdot(2l(x)-1)\,,$$

where $l(x) \in \{0,1\}$ is majority vote output of the committee consistency. For more details, we refer the reader to Prabhu et al. (2021). Additionally, at each training epoch, SENTRY balances the source data and pseudo-balances the target data. We adopted our implementation with the official implementation in Prabhu et al. (2021) with minor differences. In particular, to keep the implementation consistent with all the other DA methods, we train with the objective above from scratch instead of training sequentially after a initialization with source-only classifier as in the original paper (Prabhu et al., 2021).

Since Fix-Match, NoisyStuent, and Sentry use strong data-augmentations in their implementation, the applicability of these algorithms is restricted to vision datasets. For NLP and tabular datasets, we only train models with PseudoLabel as it doesn't rely on any augmentation technique.

F.11.4 Test-time training methods

These take an already trained source model and adapt a few parameters (e.g. batch norm parameters, batch norm statistics) on the unlabeled target data with an aim to improve target performance. Hence, we restrict these methods to vision datasets with architectures that use batch norm. These methods are computationally cheaper than other DA methods in the suite as they adapt a classifier on-the-fly. We include the following methods in our experimental suite:

BN-adapt Li et al. (2016) proposed batch norm adaptation. More recently, Schneider et al. (2020) showed gains with BN-adapt on common corruptions benchmark. Batch norm adaptation is applicable for deep models with batch norm parameters. With this method we simply adapt the Batchnorm statistics, in particular, mean and std of each batch norm layer.

TENT Wang et al. (2021a) proposed optimizing batch norm parameters to minimize the entropy of the predictor on the unlabeled target data. In our implementation of TENT, we perform BN-adapt before learning batch norm parameters.

CORAL Sun et al. (2016) proposed CORAL to adapt a model trained on the source to target by whitening the feature representations. In particular, say $\widehat{w}\Sigma_s$ is the empirical covariance of the target data representations and Σ_s is the empirical covariance of the source data representations, CORAL adjusts a linear layer g on target by re-training the final layer on the outputs: $\Sigma_t^{1/2}\Sigma_s^{-1/2}h(x)$. DARE (Rosenfeld et al., 2022) simplified the procedure and showed that this is equivalent to training a linear head h on $\Sigma_s^{-1/2}h(x)$ and whitening target data representations with $\Sigma_t^{-1/2}h(x)$ before input to the classifier. We choose to implement the latter procedure as it is cheap to train a single classifier in multi-domain datasets.

With our meta-algorithm, before adapting the source-only classifier with test time adaptation methods, we use it to perform the re-sampling correction. After obtaining the adapted classifier, we estimate target label marginal and use it to adjust the classifier with re-weighting.

F.12 Hyperparameter and Architecture Details

F.12.1 Architecture and Pretraining Details

For all datasets, we used the same architecture across different algorithms:

- CIFAR-10: Resnet-18 (He et al., 2016) pretrained on Imagenet
- CIFAR-100: Resnet-18 (He et al., 2016) pretrained on Imagenet
- Camelyon: Densenet-121 (Huang et al., 2017) *not* pretrained on Imagenet as per the suggestion made in (Koh et al., 2021)
- FMoW: Densenet-121 (Huang et al., 2017) pretrained on Imagenet
- BREEDs (Entity13, Entity30, Living17, Nonliving26): Resnet-18 (He et al., 2016) *not* pretrained on Imagenet as per the suggestion in (Santurkar et al., 2021). The main rationale is to avoid pre-training on the superset dataset where we are simulating sub-population shift.
- Officehome: Resnet-50 (He et al., 2016) pretrained on Imagenet
- Domainnet: Resnet-50 (He et al., 2016) pretrained on Imagenet
- Visda: Resnet-50 (He et al., 2016) pretrained on Imagenet
- Civilcomments: Pre-trained DistilBERT-base-uncased (Sanh et al., 2019a)
- Retiring Adults: We use an MLP with 2 hidden layers and 100 hidden units in both of the hidden layer
- Mimic Readmissions: We use the transformer architecture described in Yao et al. (2022)[1]

Except for Resnets on CIFAR datasets, we used the standard pytorch implementation (Gardner et al., 2018). For Resnet on cifar, we refer to the implementation here: `https:`

[1] https://github.com/huaxiuyao/Wild-Time/.

//github.com/kuangliu/pytorch-cifar. For all the architectures, whenever applicable, we add antialiasing (Zhang, 2019). We use the official library released with the paper.

For imagenet-pretrained models with standard architectures, we use the publicly available models here: https://pytorch.org/vision/stable/models.html. For imagenet-pretrained models on the reduced input size images (e.g. CIFAR-10), we train a model on Imagenet on reduced input size from scratch. We include the model with our publicly available repository. For bert-based models, we use the publicly available models here: https://huggingface.co/docs/transformers/.

F.12.2 Hyperparameters

First, we tune learning rate and ℓ_2 regularization parameter by fixing batch size for each dataset that correspond to maximum we can fit to 15GB GPU memory. We set the number of epochs for training as per the suggestions of the authors of respective benchmarks. Note that we define the number of epochs as a full pass over the labeled training source data. We summarize learning rate, batch size, number of epochs, and ℓ_2 regularization parameter used in our study in Table F.6.

Dataset	Epoch	Batch size	ℓ_2 regularization	Learning rate
CIFAR10	50	200	0.0001 (chosen from $\{0.0001, 0.001,$1e-5$, 0.0\}$)	0.01 (chosen from $\{0.001, 0.01, 0.0001\}$)
CIFAR100	50	200	0.0001 (chosen from $\{0.0001, 0.001,$1e-5$, 0.0\}$)	0.01 (chosen from $\{0.001, 0.01, 0.0001\}$)
Camelyon	10	96	0.01 (chosen from $\{0.01, 0.001, 0.0001, 0.0\}$)	0.03 (chosen from $\{0.003, 0.3, 0.0003, 0.03\}$)
FMoW	30	64	0.0 (chosen from $\{0.0001, 0.001,$1e-5,0.0$\}$)	0.0001 (chosen from $\{0.001, 0.01, 0.0001\}$)
Entity13	40	256	5e-5 (chosen from $\{$5e-5, 5e-4, 1e-4, 1e-5$\}$)	0.2 (chosen from $\{0.1, 0.5, 0.2, 0.01, 0.0\}$)
Entity30	40	256	5e-5 (chosen from $\{$5e-5, 5e-4, 1e-4, 1e-5$\}$)	0.2 (chosen from $\{0.1, 0.5, 0.2, 0.01, 0.0\}$)
Living17	40	256	5e-5 (chosen from $\{$5e-5, 5e-4, 1e-4, 1e-5$\}$)	0.2 (chosen from $\{0.1, 0.5, 0.2, 0.01, 0.0\}$)
Nonliving26	40	256	0 5e-5 (chosen from $\{$5e-5, 5e-4, 1e-4, 1e-5$\}$)	0.2 (chosen from $\{0.1, 0.5, 0.2, 0.01, 0.0\}$)
Officehome	50	96	0.0001 (chosen from $\{0.0001, 0.001,$1e-5$, 0.0\}$)	0.01 (chosen from $\{0.001, 0.01, 0.0001\}$)
DomainNet	15	96	0.0001 (chosen from $\{0.0001, 0.001,$1e-5$, 0.0\}$)	0.01 (chosen from $\{0.001, 0.01, 0.0001\}$)
Visda	10	96	0.0001 (chosen from $\{0.0001, 0.001,$1e-5$, 0.0\}$)	0.01 (chosen from $\{0.001, 0.01, 0.0001\}$)
Civilcomments	5	32	0.01 (chosen from $\{0.01, 0.001, 0.0001, 0.0\}$)	2e-5 (chosen from $\{2e-4, 2e-5\}$)
Retiring Adults	50	200	0.0001 (chosen from $\{0.01, 0.001, 0.0001, 0.0\}$)	0.01 (chosen from $\{0.001, 0.01, 0.0001\}$)
Mimic Readmissions	100	128	0.0 (chosen from $\{0.01, 0.001, 0.0001, 0.0\}$)	5e-4 (chosen from $\{0.005, 0.00010.0005\}$)

Table F.6: Details of the learning rate and batch size considered in our RLSBENCH

For each algorithm, we use the hyperparameters reported in the initial papers. For domain-adversarial methods (DANN and CDANN), we refer to the suggestions made in Transfer Learning Library (Jiang et al., 2022). We tabulate hyperparameters for each algorithm next:

- **DANN, CDANN, IW-CDANN and IW-DANN** As per Transfer Learning Library suggestion, we use a learning rate multiplier of 0.1 for the featurizer when initializing with a pre-trained network and 1.0 otherwise. We default to a penalty weight of 1.0 for all datasets with pre-trained initialization.

- **FixMatch** We use the lambda is 1.0 and use threshold τ as 0.9.
- **NoisyStudent** We repeat the procedure for 2 iterations and use a drop level of $p = 0.5$.
- **SENTRY** We use $\lambda_{\text{src}} = 1.0$, $\lambda_{\text{ent}} = 1.0$, and $\lambda_{\text{unsup}} = 0.1$. We use a committee of size 3.
- **PsuedoLabel** We use the lambda is 1.0 and use threshold τ as 0.9.

Recent works (Baek et al., 2022; Chen et al., 2021a; Deng and Zheng, 2021; Garg et al., 2022b; Guillory et al., 2021; Jiang et al., 2021) have proposed numerous heuristics to predict classifier performance under distribution shift. Analyzing the usefulness of these heuristics for hyperparameter selection is an interesting avenue for future work.

F.12.3 Compute Infrastructure

Our experiments were performed across a combination of Nvidia T4, A6000, P100 and V100 GPUs. Overall, to run the entire RLSBENCH suite on a T4 GPU machine with 8 CPU cores we would approximately need $70k$ GPU hours of compute.

F.12.4 Data Augmentation

In our experiments, we leverage data augmentation techniques that encourage robustness to some variations between domains for vision datasets.

For weak augmentation, we leverage random horizontal flips and random crops of pre-defined size. For strong augmentation, we apply the following transformations sequentially: random horizontal flips, random crops of pre-defined size, augmentation with Cutout (DeVries and Taylor, 2017), and RandAugment (Cubuk et al., 2020). For the exact implementation of RandAugment, we directly use the implementation of Sohn et al. (2020). The pool of operations includes: autocontrast, brightness, color jitter, contrast, equalize, posterize, rotation, sharpness, horizontal and vertical shearing, solarize, and horizontal and vertical translations. We apply N = 2 random operations for all experiments.

Appendix G

Appendix: RATT: Leveraging Unlabeled Data to Guarantee Generalization

Throughout this discussion, we will make frequently use of the following standard results concerning the exponential concentration of random variables:

Lemma G.0.1 (Hoeffding's inequality for independent RVs (Hoeffding, 1994))**.** *Let $Z_1, Z_2, \ldots, Z_n$ be independent bounded random variables with $Z_i \in [a, b]$ for all i, then*

$$\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^{n}(Z_i - \mathbb{E}\left[Z_i\right]) \geqslant t\right) \leqslant \exp\left(-\frac{2nt^2}{(b-a)^2}\right)$$

and

$$\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^{n}(Z_i - \mathbb{E}\left[Z_i\right]) \leqslant -t\right) \leqslant \exp\left(-\frac{2nt^2}{(b-a)^2}\right)$$

for all $t \geqslant 0$.

Lemma G.0.2 (Hoeffding's inequality for sampling with replacement (Hoeffding, 1994))**.** *Let $\mathcal{Z} = (Z_1, Z_2, \ldots, Z_N)$ be a finite population of N points with $Z_i \in [a.b]$ for all i. Let $X_1, X_2, \ldots X_n$ be a random sample drawn without replacement from $\mathcal{Z}$. Then for all $t \geqslant 0$, we have*

$$\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^{n}(X_i - \mu) \geqslant t\right) \leqslant \exp\left(-\frac{2nt^2}{(b-a)^2}\right)$$

and

$$\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^{n}(X_i - \mu) \leqslant -t\right) \leqslant \exp\left(-\frac{2nt^2}{(b-a)^2}\right),$$

where $\mu = \frac{1}{N}\sum_{i=1}^{N} Z_i$.

We now discuss one condition that generalizes the exponential concentration to dependent random variables.

Condition G.0.3 (Bounded difference inequality)**.** *Let $\mathcal{Z}$ be some set and $\phi : \mathcal{Z}^n \to \mathbb{R}$. We say that ϕ satisfies the bounded difference assumption if there exists $c_1, c_2, \dots c_n \geqslant 0$ s.t. for all i, we have*

$$\sup_{Z_1, Z_2, \dots, Z_n, Z_i' \in \mathcal{Z}^{n+1}} |\phi(Z_1, \dots, Z_i, \dots, Z_n) - \phi(Z_1, \dots, Z_i', \dots, Z_n)| \leqslant c_i \,.$$

Lemma G.0.4 (McDiarmid's inequality (McDiarmid, 1989))**.** *Let $Z_1, Z_2, \dots, Z_n$ be independent random variables on set $\mathcal{Z}$ and $\phi : \mathcal{Z}^n \to \mathbb{R}$ satisfy bounded difference inequality (Condition G.0.3). Then for all $t > 0$, we have*

$$\mathbb{P}\left(\phi(Z_1, Z_2, \dots, Z_n) - \mathbb{E}\left[\phi(Z_1, Z_2, \dots, Z_n)\right] \geqslant t\right) \leqslant \exp\left(-\frac{2t^2}{\sum_{i=1}^n c_i^2}\right)$$

and

$$\mathbb{P}\left(\phi(Z_1, Z_2, \dots, Z_n) - \mathbb{E}\left[\phi(Z_1, Z_2, \dots, Z_n)\right] \leqslant -t\right) \leqslant \exp\left(-\frac{2t^2}{\sum_{i=1}^n c_i^2}\right) .$$

G.1 Proofs from Sec. 8.3

Additional notation Let m_1 be the number of mislabeled points ($\widetilde{S}_M$) and m_2 be the number of correctly labeled points ($\widetilde{S}_C$). Note $m_1 + m_2 = m$.

G.1.1 Proof of Theorem 8.3.1

Proof of Lemma 8.3.2. The main idea of our proof is to regard the clean portion of the data ($S \cup \widetilde{S}_C$) as fixed. Then, there exists an (unknown) classifier f^* that minimizes the expected risk calculated on the (fixed) clean data and (random draws of) the mislabeled data $\widetilde{S}_M$. Formally,

$$f^* := \operatorname*{arg\,min}_{f \in \mathcal{F}} \mathcal{E}_{\check{\mathcal{D}}}(f) \,, \tag{G.1}$$

where

$$\check{\mathcal{D}} = \frac{n}{m+n}\mathcal{S} + \frac{m_2}{m+n}\widetilde{\mathcal{S}}_C + \frac{m_1}{m+n}\mathcal{D}' \,.$$

Note here that $\check{\mathcal{D}}$ is a combination of the *empirical distribution* over correctly labeled data $S \cup \widetilde{S}_C$ and the (population) distribution over mislabeled data $\mathcal{D}'$. Recall that

$$\widehat{w}f := \operatorname*{arg\,min}_{f \in \mathcal{F}} \mathcal{E}_{\mathcal{S} \cup \widetilde{\mathcal{S}}}(f) \,. \tag{G.2}$$

Since, $\widehat{f}$ minimizes 0-1 error on $S \cup \widetilde{S}$, using ERM optimality on (G.2), we have

$$\mathcal{E}_{\mathcal{S} \cup \widetilde{\mathcal{S}}}(\widehat{f}) \leqslant \mathcal{E}_{\mathcal{S} \cup \widetilde{\mathcal{S}}}(f^*) \,. \tag{G.3}$$

Moreover, since f^* is independent of $\widetilde{S}_M$, using Hoeffding's bound, we have with probability at least $1-\delta$ that

$$\mathcal{E}_{\widetilde{\mathcal{S}}_M}(f^*) \leqslant \mathcal{E}_{\mathcal{D}'}(f^*) + \sqrt{\frac{\log(1/\delta)}{2m_1}} \, . \tag{G.4}$$

Finally, since f^* is the optimal classifier on $\breve{\mathcal{D}}$, we have

$$\mathcal{E}_{\breve{\mathcal{D}}}(f^*) \leqslant \mathcal{E}_{\breve{\mathcal{D}}}(\widehat{f}) \, . \tag{G.5}$$

Now to relate (H.20) and (H.22), we multiply (H.21) by $\frac{m_1}{m+n}$ and add $\frac{n}{m+n}\mathcal{E}_{\mathcal{S}}(f) + \frac{m_2}{m+n}\mathcal{E}_{\widetilde{\mathcal{S}}_C}(f)$ both the sides. Hence, we can rewrite (H.21) as follows:

$$\mathcal{E}_{\mathcal{S}\cup\widetilde{\mathcal{S}}}(f^*) \leqslant \mathcal{E}_{\breve{\mathcal{D}}}(f^*) + \frac{m_1}{m+n}\sqrt{\frac{\log(1/\delta)}{2m_1}} \, . \tag{G.6}$$

Now we combine equations (H.20), (G.6), and (H.22), to get

$$\mathcal{E}_{\mathcal{S}\cup\widetilde{\mathcal{S}}}(\widehat{w}f) \leqslant \mathcal{E}_{\breve{\mathcal{D}}}(\widehat{w}f) + \frac{m_1}{m+n}\sqrt{\frac{\log(1/\delta)}{2m_1}} \, , \tag{G.7}$$

which implies

$$\mathcal{E}_{\widetilde{\mathcal{S}}_M}(\widehat{w}f) \leqslant \mathcal{E}_{\mathcal{D}'}(\widehat{w}f) + \sqrt{\frac{\log(1/\delta)}{2m_1}} \, . \tag{G.8}$$

Since $\widetilde{S}$ is obtained by randomly labeling an unlabeled dataset, we assume $2m_1 \approx m$ [1]. Moreover, using $\mathcal{E}_{\mathcal{D}'} = 1 - \mathcal{E}_{\mathcal{D}}$ we obtain the desired result. □

Proof of Lemma 8.3.3. Recall $\mathcal{E}_{\widetilde{\mathcal{S}}}(f) = \frac{m_1}{m}\mathcal{E}_{\widetilde{\mathcal{S}}_M}(f) + \frac{m_2}{m}\mathcal{E}_{\widetilde{\mathcal{S}}_C}(f)$. Hence, we have

$$2\mathcal{E}_{\widetilde{\mathcal{S}}}(f) - \mathcal{E}_{\widetilde{\mathcal{S}}_M}(f) - \mathcal{E}_{\widetilde{\mathcal{S}}_C}(f) = \left(\frac{2m_1}{m}\mathcal{E}_{\widetilde{\mathcal{S}}_M}(f) - \mathcal{E}_{\widetilde{\mathcal{S}}_M}(f)\right) + \left(\frac{2m_2}{m}\mathcal{E}_{\widetilde{\mathcal{S}}_C}(f) - \mathcal{E}_{\widetilde{\mathcal{S}}_C}(f)\right) \tag{G.9}$$

$$= \left(\frac{2m_1}{m} - 1\right)\mathcal{E}_{\widetilde{\mathcal{S}}_M}(f) + \left(\frac{2m_2}{m} - 1\right)\mathcal{E}_{\widetilde{\mathcal{S}}_C}(f) \, . \tag{G.10}$$

Since the dataset is labeled uniformly at random, with probability at least $1-\delta$, we have $\left(\frac{2m_1}{m} - 1\right) \leqslant \sqrt{\frac{\log(1/\delta)}{2m}}$. Similarly, we have with probability at least $1-\delta$, $\left(\frac{2m_2}{m} - 1\right) \leqslant \sqrt{\frac{\log(1/\delta)}{2m}}$. Using union bound, with probability at least $1-\delta$, we have

$$2\mathcal{E}_{\widetilde{\mathcal{S}}} - \mathcal{E}_{\widetilde{\mathcal{S}}_M}(f) - \mathcal{E}_{\widetilde{\mathcal{S}}_C}(f) \leqslant \sqrt{\frac{\log(2/\delta)}{2m}}\left(\mathcal{E}_{\widetilde{\mathcal{S}}_M}(f) + \mathcal{E}_{\widetilde{\mathcal{S}}_C}(f)\right) \, . \tag{G.11}$$

[1] Formally, with probability at least $1-\delta$, we have $(m - 2m_1) \leqslant \sqrt{m\log(1/\delta)/2}$.

With re-arranging $\mathcal{E}_{\widetilde{S}_M}(f) + \mathcal{E}_{\widetilde{S}_C}(f)$ and using the inequality $1 - a \leqslant \frac{1}{1+a}$, we have

$$2\mathcal{E}_{\widetilde{S}} - \mathcal{E}_{\widetilde{S}_M}(f) - \mathcal{E}_{\widetilde{S}_C}(f) \leqslant 2\mathcal{E}_{\widetilde{S}}\sqrt{\frac{\log(2/\delta)}{2m}}\,. \tag{G.12}$$

□

Proof of Lemma 8.3.4. In the set of correctly labeled points $S \cup \widetilde{S}_C$, we have S as a random subset of $S \cup \widetilde{S}_C$. Hence, using Hoeffding's inequality for sampling without replacement (Lemma G.0.2), we have with probability at least $1 - \delta$

$$\mathcal{E}_{\widetilde{S}_C}(\widehat{w}f) - \mathcal{E}_{\mathcal{S}\cup\widetilde{S}_C}(\widehat{w}f) \leqslant \sqrt{\frac{\log(1/\delta)}{2m_2}}\,. \tag{G.13}$$

Re-writing $\mathcal{E}_{\mathcal{S}\cup\widetilde{S}_C}(\widehat{w}f)$ as $\frac{m_2}{m_2+n}\mathcal{E}_{\widetilde{S}_C}(\widehat{w}f) + \frac{n}{m_2+n}\mathcal{E}_{\mathcal{S}}(\widehat{w}f)$, we have with probability at least $1 - \delta$

$$\left(\frac{n}{n+m_2}\right)\left(\mathcal{E}_{\widetilde{S}_C}(\widehat{w}f) - \mathcal{E}_{\mathcal{S}}(\widehat{w}f)\right) \leqslant \sqrt{\frac{\log(1/\delta)}{2m_2}}\,. \tag{G.14}$$

As before, assuming $2m_2 \approx m$, we have with probability at least $1 - \delta$

$$\mathcal{E}_{\widetilde{S}_C}(\widehat{w}f) - \mathcal{E}_{\mathcal{S}}(\widehat{w}f) \leqslant \left(1 + \frac{m_2}{n}\right)\sqrt{\frac{\log(1/\delta)}{m}} \leqslant \left(1 + \frac{m}{2n}\right)\sqrt{\frac{\log(1/\delta)}{m}}\,. \tag{G.15}$$

□

Proof of Theorem 8.3.1. Having established these core intermediate results, we can now combine above three lemmas to prove the main result. In particular, we bound the population error on clean data ($\mathcal{E}_{\mathcal{D}}(\widehat{w}f)$) as follows:

(i) First, use (G.8), to obtain an upper bound on the population error on clean data, i.e., with probability at least $1 - \delta/4$, we have

$$\mathcal{E}_{\mathcal{D}}(\widehat{w}f) \leqslant 1 - \mathcal{E}_{\widetilde{S}_M}(\widehat{w}f) + \sqrt{\frac{\log(4/\delta)}{m}}\,. \tag{G.16}$$

(ii) Second, use (G.12), to relate the error on the mislabeled fraction with error on clean portion of randomly labeled data and error on whole randomly labeled dataset, i.e., with probability at least $1 - \delta/2$, we have

$$-\mathcal{E}_{\widetilde{S}_M}(f) \leqslant \mathcal{E}_{\widetilde{S}_C}(f) - 2\mathcal{E}_{\widetilde{S}} + 2\mathcal{E}_{\widetilde{S}}\sqrt{\frac{\log(4/\delta)}{2m}}\,. \tag{G.17}$$

(iii) Finally, use (G.15) to relate the error on the clean portion of randomly labeled data and error on clean training data, i.e., with probability $1 - \delta/4$, we have

$$\mathcal{E}_{\widetilde{\mathcal{S}}_C}(\widehat{w} f) \leqslant -\mathcal{E}_{\mathcal{S}}(\widehat{w} f) + \left(1 + \frac{m}{2n}\right)\sqrt{\frac{\log(4/\delta)}{m}} \,. \tag{G.18}$$

Using union bound on the above three steps, we have with probability at least $1 - \delta$:

$$\mathcal{E}_{\mathcal{D}}(\widehat{w} f) \leqslant \mathcal{E}_{\mathcal{S}}(\widehat{w} f) + 1 - 2\mathcal{E}_{\widetilde{\mathcal{S}}}(\widehat{w} f) + \left(\sqrt{2}\mathcal{E}_{\widetilde{\mathcal{S}}} + 2 + \frac{m}{2n}\right)\sqrt{\frac{\log(4/\delta)}{m}} \,. \tag{G.19}$$

□

G.1.2 Proof of Proposition 8.3.5

Proof of Proposition 8.3.5. For a classifier $f : \mathcal{X} \to \{-1, 1\}$, we have $1 - 2\,\mathbb{I}\left[f(x) \neq y\right] = y \cdot f(x)$. Hence, by definition of $\mathcal{E}$, we have

$$1 - 2\mathcal{E}_{\widetilde{\mathcal{S}}}(f) = \frac{1}{m}\sum_{i=1}^{m} y_i \cdot f(x_i) \leqslant \sup_{f \in \mathcal{F}} \frac{1}{m}\sum_{i=1}^{m} y_i \cdot f(x_i) \,. \tag{G.20}$$

Note that for fixed inputs $(x_1, x_2, \ldots, x_m)$ in $\widetilde{S}$, $(y_1, y_2, \ldots y_m)$ are random labels. Define $\phi_1(y_1, y_2, \ldots, y_m) := \sup_{f \in \mathcal{F}} \frac{1}{m}\sum_{i=1}^{m} y_i \cdot f(x_i)$. We have the following bounded difference condition on ϕ_1. For all i,

$$\sup_{y_1, \ldots y_m, y_i' \in \{-1,1\}^{m+1}} \left|\phi_1(y_1, \ldots, y_i, \ldots, y_m) - \phi_1(y_1, \ldots, y_i', \ldots, y_m)\right| \leqslant 1/m \,. \tag{G.21}$$

Similarly, we define $\phi_2(x_1, x_2, \ldots, x_m) := \mathbb{E}_{y_i \sim U\{-1,1\}}\left[\sup_{f \in \mathcal{F}} \frac{1}{m}\sum_{i=1}^{m} y_i \cdot f(x_i)\right]$. We have the following bounded difference condition on ϕ_2. For all i,

$$\sup_{x_1, \ldots x_m, x_i' \in \mathcal{X}^{m+1}} \left|\phi_2(x_1, \ldots, x_i, \ldots, x_m) - \phi_1(x_1, \ldots, x_i', \ldots, x_m)\right| \leqslant 1/m \,. \tag{G.22}$$

Using McDiarmid's inequality (Lemma G.0.4) twice with Condition (G.21) and (G.22), with probability at least $1 - \delta$, we have

$$\sup_{f \in \mathcal{F}} \frac{1}{m}\sum_{i=1}^{m} y_i \cdot f(x_i) - \mathbb{E}_{x,y}\left[\sup_{f \in \mathcal{F}} \frac{1}{m}\sum_{i=1}^{m} y_i \cdot f(x_i)\right] \leqslant \sqrt{\frac{2\log(2/\delta)}{m}} \,. \tag{G.23}$$

Combining (G.20) and (G.23), we obtain the desired result. □

G.1.3 Proof of Theorem 8.3.6

Proof of Theorem 8.3.6 follows similar to the proof of Theorem 8.3.1. Note that the same results in Lemma 8.3.2, Lemma 8.3.3, and Lemma 8.3.4 hold in the regularized ERM case. However, the arguments in the proof of Lemma 8.3.2 change slightly. Hence, we state the lemma for regularized ERM and prove it here for completeness.

Lemma G.1.1. *Assume the same setup as Theorem 8.3.6. Then for any $\delta > 0$, with probability at least $1-\delta$ over the random draws of mislabeled data $\widetilde{S}_M$, we have*

$$\mathcal{E}_{\mathcal{D}}(\widehat{f}) \leqslant 1 - \mathcal{E}_{\widetilde{\mathcal{S}}_M}(\widehat{f}) + \sqrt{\frac{\log(1/\delta)}{m}}\,. \tag{G.24}$$

Proof. The main idea of the proof remains the same, i.e. regard the clean portion of the data ($S \cup \widetilde{S}_C$) as fixed. Then, there exists a classifier f^* that is optimal over draws of the mislabeled data $\widetilde{S}_M$.

Formally,

$$f^* := \arg\min_{f \in \mathcal{F}} \mathcal{E}_{\check{\mathcal{D}}}(f) + \lambda R(f)\,, \tag{G.25}$$

where

$$\check{\mathcal{D}} = \frac{n}{m+n}\mathcal{S} + \frac{m_1}{m+n}\widetilde{\mathcal{S}}_C + \frac{m_2}{m+n}\mathcal{D}'\,.$$

That is, $\check{\mathcal{D}}$ a combination of the *empirical distribution* over correctly labeled data $S \cup \widetilde{S}_C$ and the (population) distribution over mislabeled data $\mathcal{D}'$. Recall that

$$\hat{w}f := \arg\min_{f \in \mathcal{F}} \mathcal{E}_{\mathcal{S}\cup\widetilde{\mathcal{S}}}(f) + \lambda R(f)\,. \tag{G.26}$$

Since, $\widehat{f}$ minimizes 0-1 error on $S \cup \widetilde{S}$, using ERM optimality on (G.2), we have

$$\mathcal{E}_{\mathcal{S}\cup\widetilde{\mathcal{S}}}(\widehat{f}) + \lambda R(\hat{w}f) \leqslant \mathcal{E}_{\mathcal{S}\cup\widetilde{\mathcal{S}}}(f^*) + \lambda R(f^*)\,. \tag{G.27}$$

Moreover, since f^* is independent of $\widetilde{S}_M$, using Hoeffding's bound, we have with probability at least $1-\delta$ that

$$\mathcal{E}_{\widetilde{\mathcal{S}}_M}(f^*) \leqslant \mathcal{E}_{\mathcal{D}'}(f^*) + \sqrt{\frac{\log(1/\delta)}{2m_1}}\,. \tag{G.28}$$

Finally, since f^* is the optimal classifier on $\check{\mathcal{D}}$, we have

$$\mathcal{E}_{\check{\mathcal{D}}}(f^*) + \lambda R(f^*) \leqslant \mathcal{E}_{\check{\mathcal{D}}}(\widehat{f}) + \lambda R(\hat{w}f)\,. \tag{G.29}$$

Now to relate (G.27) and (G.29), we can re-write the (G.28) as follows:

$$\mathcal{E}_{\mathcal{S}\cup\widetilde{\mathcal{S}}}(f^*) \leqslant \mathcal{E}_{\check{\mathcal{D}}}(f^*) + \frac{m_1}{m+n}\sqrt{\frac{\log(1/\delta)}{2m_1}}\,. \tag{G.30}$$

After adding $\lambda R(f^*)$ on both sides in (G.30), we combine equations (G.27), (G.30), and (G.29), to get

$$\mathcal{E}_{\mathcal{S}\cup\tilde{\mathcal{S}}}(\widehat{w}f) \leqslant \mathcal{E}_{\check{\mathcal{D}}}(\widehat{w}f) + \frac{m_1}{m+n}\sqrt{\frac{\log(1/\delta)}{2m_1}}\,, \tag{G.31}$$

which implies

$$\mathcal{E}_{\tilde{\mathcal{S}}_M}(\widehat{w}f) \leqslant \mathcal{E}_{\mathcal{D}'}(\widehat{w}f) + \sqrt{\frac{\log(1/\delta)}{2m_1}}\,. \tag{G.32}$$

Similar as before, since $\widetilde{S}$ is obtained by randomly labeling an unlabeled dataset, we assume $2m_1 \approx m$. Moreover, using $\mathcal{E}_{\mathcal{D}'} = 1 - \mathcal{E}_{\mathcal{D}}$ we obtain the desired result. □

G.1.4 Proof of Theorem 8.3.7

To prove our results in the multiclass case, we first state and prove lemmas parallel to those used in the proof of balanced binary case. We then combine these results to obtain the result in Theorem 8.3.7.

Before stating the result, we define mislabeled distribution $\mathcal{D}'$ for any $\mathcal{D}$. While $\mathcal{D}'$ and $\mathcal{D}$ share the same marginal distribution over inputs $\mathcal{X}$, the conditional distribution over labels y given an input $x \sim \mathcal{D}_{\mathcal{X}}$ is changed as follows: For any x, the Probability Mass Function (PMF) over y is defined as: $p_{\mathcal{D}'}(\cdot|x) := \frac{1-p_{\mathcal{D}}(\cdot|x)}{k-1}$, where $p_{\mathcal{D}}(\cdot|x)$ is the PMF over y for the distribution $\mathcal{D}$.

Lemma G.1.2. *Assume the same setup as Theorem 8.3.7. Then for any $\delta > 0$, with probability at least $1-\delta$ over the random draws of mislabeled data $\widetilde{S}_M$, we have*

$$\mathcal{E}_{\mathcal{D}}(\widehat{f}) \leqslant (k-1)\left(1 - \mathcal{E}_{\tilde{\mathcal{S}}_M}(\widehat{f})\right) + (k-1)\sqrt{\frac{\log(1/\delta)}{m}}\,. \tag{G.33}$$

Proof. The main idea of the proof remains the same. We begin by regarding the clean portion of the data $(S \cup \widetilde{S}_C)$ as fixed. Then, there exists a classifier f^* that is optimal over draws of the mislabeled data $\widetilde{S}_M$.

However, in the multiclass case, we cannot as easily relate the population error on mislabeled data to the population accuracy on clean data. While for binary classification, we could lower bound the population accuracy $1 - \mathcal{E}_{\mathcal{D}}$ with the empirical error on mislabeled data $\mathcal{E}_{\tilde{\mathcal{S}}_M}$ (in the proof of Lemma 8.3.2), for multiclass classification, error on the mislabeled data and accuracy on the clean data in the population are not so directly related. To establish (G.33), we break the error on the (unknown) mislabeled data into two parts: one term corresponds to predicting the true label on mislabeled data, and the other corresponds to predicting neither the true label nor the assigned (mis-)label. Finally, we relate these errors to their population counterparts to establish (G.33).

Formally,

$$f^* := \arg\min_{f\in\mathcal{F}} \mathcal{E}_{\breve{\mathcal{D}}}(f) + \lambda R(f)\,, \tag{G.34}$$

where

$$\breve{\mathcal{D}} = \frac{n}{m+n}\mathcal{S} + \frac{m_1}{m+n}\widetilde{\mathcal{S}}_C + \frac{m_2}{m+n}\mathcal{D}'\,.$$

That is, $\breve{\mathcal{D}}$ is a combination of the *empirical distribution* over correctly labeled data $S \cup \widetilde{S}_C$ and the (population) distribution over mislabeled data $\mathcal{D}'$. Recall that

$$\widehat{w}f := \arg\min_{f\in\mathcal{F}} \mathcal{E}_{\mathcal{S}\cup\widetilde{\mathcal{S}}}(f) + \lambda R(f)\,. \tag{G.35}$$

Following the exact steps from the proof of Lemma G.1.1, with probability at least $1-\delta$, we have

$$\mathcal{E}_{\widetilde{\mathcal{S}}_M}(\widehat{w}f) \leqslant \mathcal{E}_{\mathcal{D}'}(\widehat{w}f) + \sqrt{\frac{\log(1/\delta)}{2m_1}}\,. \tag{G.36}$$

Similar to before, since $\widetilde{S}$ is obtained by randomly labeling an unlabeled dataset, we assume $\frac{k}{k-1}m_1 \approx m$.

Now we will relate $\mathcal{E}_{\mathcal{D}'}(\widehat{w}f)$ with $\mathcal{E}_{\mathcal{D}}(\widehat{w}f)$. Let y^T denote the (unknown) true label for a mislabeled point (x, y) (i.e., label before replacing it with a mislabel).

$$\begin{aligned}\mathbb{E}_{(x,y)\in\sim\mathcal{D}'}\left[\mathbb{I}\left[\widehat{w}f(x) \neq y\right]\right] = \underbrace{\mathbb{E}_{(x,y)\in\sim\mathcal{D}'}\left[\mathbb{I}\left[\widehat{w}f(x) \neq y \wedge \widehat{w}f(x) \neq y^T\right]\right]}_{\mathrm{I}} \\ + \underbrace{\mathbb{E}_{(x,y)\in\sim\mathcal{D}'}\left[\mathbb{I}\left[\widehat{w}f(x) \neq y \wedge \widehat{w}f(x) = y^T\right]\right]}_{\mathrm{II}}\,.\end{aligned} \tag{G.37}$$

Clearly, term 2 is one minus the accuracy on the clean unseen data, i.e.,

$$\mathrm{II} = 1 - \mathbb{E}_{x,y\sim\mathcal{D}}\left[\mathbb{I}\left[\widehat{w}f(x) \neq y\right]\right] = 1 - \mathcal{E}_{\mathcal{D}}(\widehat{w}f)\,. \tag{G.38}$$

Next, we relate term 1 with the error on the unseen clean data. We show that term 1 is equal to the error on the unseen clean data scaled by $\frac{k-2}{k-1}$, where k is the number of labels. Using the definition of mislabeled distribution $\mathcal{D}'$, we have

$$\mathrm{I} = \frac{1}{k-1}\left(\mathbb{E}_{(x,y)\in\sim\mathcal{D}}\left[\sum_{i\in\mathcal{Y}\wedge i\neq y} \mathbb{I}\left[\widehat{w}f(x) \neq i \wedge \widehat{w}f(x) \neq y\right]\right]\right) = \frac{k-2}{k-1}\mathcal{E}_{\mathcal{D}}(\widehat{w}f)\,. \tag{G.39}$$

Combining the result in (G.38), (G.39) and (G.37), we have

$$\mathcal{E}_{\mathcal{D}'}(\widehat{w}f) = 1 - \frac{1}{k-1}\mathcal{E}_{\mathcal{D}}(\widehat{w}f)\,. \tag{G.40}$$

Finally, combining the result in (G.40) with equation (G.36), we have with probability $1-\delta$,

$$\mathcal{E}_{\mathcal{D}}(\widehat{w}f) \leqslant (k-1)\left(1-\mathcal{E}_{\widetilde{S}_M}(\widehat{w}f)\right) + (k-1)\sqrt{\frac{k\log(1/\delta)}{2(k-1)m}}\,. \tag{G.41}$$

□

Lemma G.1.3. *Assume the same setup as Theorem 8.3.7. Then for any $\delta > 0$, with probability at least $1-\delta$ over the random draws of $\widetilde{S}$, we have*

$$\left|k\mathcal{E}_{\widetilde{S}}(\widehat{f}) - \mathcal{E}_{\widetilde{S}_C}(\widehat{f}) - (k-1)\mathcal{E}_{\widetilde{S}_M}(\widehat{f})\right| \leqslant 2k\sqrt{\frac{\log(4/\delta)}{2m}}\,.$$

Proof. Recall $\mathcal{E}_{\widetilde{S}}(f) = \frac{m_1}{m}\mathcal{E}_{\widetilde{S}_M}(f) + \frac{m_2}{m}\mathcal{E}_{\widetilde{S}_C}(f)$. Hence, we have

$$\begin{aligned} k\mathcal{E}_{\widetilde{S}}(f) - (k-1)\mathcal{E}_{\widetilde{S}_M}(f) - \mathcal{E}_{\widetilde{S}_C}(f) &= (k-1)\left(\frac{km_1}{(k-1)m}\mathcal{E}_{\widetilde{S}_M}(f) - \mathcal{E}_{\widetilde{S}_M}(f)\right) \\ &\quad + \left(\frac{km_2}{m}\mathcal{E}_{\widetilde{S}_C}(f) - \mathcal{E}_{\widetilde{S}_C}(f)\right) \\ &= k\left[\left(\frac{m_1}{m} - \frac{k-1}{k}\right)\mathcal{E}_{\widetilde{S}_M}(f) + \left(\frac{m_2}{m} - \frac{1}{k}\right)\mathcal{E}_{\widetilde{S}_C}(f)\right]. \end{aligned}$$

Since the dataset is randomly labeled, we have with probability at least $1-\delta$, $\left(\frac{m_1}{m} - \frac{k-1}{k}\right) \leqslant \sqrt{\frac{\log(1/\delta)}{2m}}$. Similarly, we have with probability at least $1-\delta$, $\left(\frac{m_2}{m} - \frac{1}{k}\right) \leqslant \sqrt{\frac{\log(1/\delta)}{2m}}$. Using union bound, we have with probability at least $1-\delta$

$$k\mathcal{E}_{\widetilde{S}}(f) - (k-1)\mathcal{E}_{\widetilde{S}_M}(f) - \mathcal{E}_{\widetilde{S}_C}(f) \leqslant k\sqrt{\frac{\log(2/\delta)}{2m}}\left(\mathcal{E}_{\widetilde{S}_M}(f) + \mathcal{E}_{\widetilde{S}_C}(f)\right). \tag{G.42}$$

□

Lemma G.1.4. *Assume the same setup as Theorem 8.3.7. Then for any $\delta > 0$, with probability at least $1-\delta$ over the random draws of $\widetilde{S}_C$ and S, we have*

$$\left|\mathcal{E}_{\widetilde{S}_C}(\widehat{f}) - \mathcal{E}_{\mathcal{S}}(\widehat{f})\right| \leqslant 1.5\sqrt{\frac{k\log(2/\delta)}{2m}}\,.$$

Proof. In the set of correctly labeled points $S \cup \widetilde{S}_C$, we have S as a random subset of $S \cup \widetilde{S}_C$. Hence, using Hoeffding's inequality for sampling without replacement (Lemma G.0.2), we have with probability at least $1-\delta$

$$\mathcal{E}_{\widetilde{S}_c}(\widehat{w}f) - \mathcal{E}_{\mathcal{S}\cup\widetilde{S}_C}(\widehat{w}f) \leqslant \sqrt{\frac{\log(1/\delta)}{2m_2}}\,. \tag{G.43}$$

Re-writing $\mathcal{E}_{\mathcal{S}\cup\widetilde{\mathcal{S}}_C}(\widehat{w}f)$ as $\frac{m_2}{m_2+n}\mathcal{E}_{\widetilde{\mathcal{S}}_C}(\widehat{w}f) + \frac{n}{m_2+n}\mathcal{E}_{\mathcal{S}}(\widehat{w}f)$, we have with probability at least $1-\delta$

$$\left(\frac{n}{n+m_2}\right)\left(\mathcal{E}_{\widetilde{\mathcal{S}}_c}(\widehat{w}f) - \mathcal{E}_{\mathcal{S}}(\widehat{w}f)\right) \leqslant \sqrt{\frac{\log(1/\delta)}{2m_2}}\,. \tag{G.44}$$

As before, assuming $km_2 \approx m$, we have with probability at least $1-\delta$

$$\mathcal{E}_{\widetilde{\mathcal{S}}_c}(\widehat{w}f) - \mathcal{E}_{\mathcal{S}}(\widehat{w}f) \leqslant \left(1+\frac{m_2}{n}\right)\sqrt{\frac{k\log(1/\delta)}{2m}} \leqslant \left(1+\frac{1}{k}\right)\sqrt{\frac{k\log(1/\delta)}{2m}}\,. \tag{G.45}$$

□

Proof of Theorem 8.3.7. Having established these core intermediate results, we can now combine above three lemmas. In particular, we bound the population error on clean data ($\mathcal{E}_{\mathcal{D}}(\widehat{w}f)$) as follows:

(i) First, use (G.41), to obtain an upper bound on the population error on clean data, i.e., with probability at least $1-\delta/4$, we have

$$\mathcal{E}_{\mathcal{D}}(\widehat{w}f) \leqslant (k-1)\left(1-\mathcal{E}_{\widetilde{\mathcal{S}}_M}(\widehat{w}f)\right) + (k-1)\sqrt{\frac{k\log(4/\delta)}{2(k-1)m}}\,. \tag{G.46}$$

(ii) Second, use (G.42) to relate the error on the mislabeled fraction with error on clean portion of randomly labeled data and error on whole randomly labeled dataset, i.e., with probability at least $1-\delta/2$, we have

$$-(k-1)\mathcal{E}_{\widetilde{\mathcal{S}}_M}(f) \leqslant \mathcal{E}_{\widetilde{\mathcal{S}}_C}(f) - k\mathcal{E}_{\widetilde{\mathcal{S}}} + k\sqrt{\frac{\log(4/\delta)}{2m}}\,. \tag{G.47}$$

(iii) Finally, use (G.45) to relate the error on the clean portion of randomly labeled data and error on clean training data, i.e., with probability $1-\delta/4$, we have

$$\mathcal{E}_{\widetilde{\mathcal{S}}_C}(\widehat{w}f) \leqslant -\mathcal{E}_{\mathcal{S}}(\widehat{w}f) + \left(1+\frac{m}{kn}\right)\sqrt{\frac{k\log(4/\delta)}{2m}}\,. \tag{G.48}$$

Using union bound on the above three steps, we have with probability at least $1-\delta$:

$$\mathcal{E}_{\mathcal{D}}(\widehat{w}f) \leqslant \mathcal{E}_{\mathcal{S}}(\widehat{w}f) + (k-1) - k\mathcal{E}_{\widetilde{\mathcal{S}}}(\widehat{w}f) + (\sqrt{k(k-1)} + k + \sqrt{k} + \frac{m}{n\sqrt{k}})\sqrt{\frac{\log(4/\delta)}{2m}}\,. \tag{G.49}$$

Simplifying the term in RHS of (G.49), we get the desired result. in the final bound. □

G.2 Proofs from Sec. 8.4

We suppose that the parameters of the linear function are obtained via gradient descent on the following L_2 regularized problem:

$$\mathcal{L}_S(w;\lambda) := \sum_{i=1}^{n} (w^T x_i - y_i)^2 + \lambda \left\| w \right\| 2^2 \,, \tag{G.50}$$

where $\lambda \geqslant 0$ is a regularization parameter. We assume access to a clean dataset $S = \{(x_i, y_i)\}_{i=1}^n \sim \mathcal{D}^n$ and randomly labeled dataset $\widetilde{S} = \{(x_i, y_i)\}_{i=n+1}^{n+m} \sim \widetilde{\mathcal{D}}^m$. Let $\boldsymbol{X} = [x_1, x_2, \cdots, x_{m+n}]$ and $\boldsymbol{y} = [y_1, y_2, \cdots, y_{m+n}]$. Fix a positive learning rate η such that $\eta \leqslant 1/\left(\left\|\boldsymbol{X}^T\boldsymbol{X}\right\| \mathrm{op} + \lambda^2\right)$ and an initialization $w_0 = 0$. Consider the following gradient descent iterates to minimize objective (G.50) on $S \cup \widetilde{S}$:

$$w_t = w_{t-1} - \eta \nabla_w \mathcal{L}_{S\cup\widetilde{S}}(w_{t-1};\lambda) \quad \forall t = 1, 2, \ldots \tag{G.51}$$

Then we have $\{w_t\}$ converge to the limiting solution $\widehat{w}w = \left(\boldsymbol{X}^T\boldsymbol{X} + \lambda \boldsymbol{I}\right)^{-1} \boldsymbol{X}^T\boldsymbol{y}$. Define $\widehat{f}(x) := f(x; \widehat{w}w)$.

G.2.1 Proof of Theorem 8.4.2

We use a standard result from linear algebra, namely the Shermann-Morrison formula (Sherman and Morrison, 1950) for matrix inversion:

Lemma G.2.1 (Sherman and Morrison (1950))**.** *Suppose $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ is an invertible square matrix and $u, v \in \mathbb{R}^n$ are column vectors. Then $\boldsymbol{A} + uv^T$ is invertible iff $1 + v^T\boldsymbol{A}u \neq 0$ and in particular*

$$(\boldsymbol{A} + uv^T)^{-1} = \boldsymbol{A}^{-1} - \frac{\boldsymbol{A}^{-1}uv^T\boldsymbol{A}^{-1}}{1 + v^T\boldsymbol{A}^{-1}u} \,. \tag{G.52}$$

For a given training set $S \cup \widetilde{S}_C$, define leave-one-out error on mislabeled points in the training data as

$$\mathcal{E}_{\mathrm{LOO}(\widetilde{S}_M)} = \frac{\sum_{(x_i,y_i)\in\widetilde{S}_M} \mathcal{E}(f_{(i)}(x_i), y_i)}{\left|\widetilde{S}_M\right|} \,,$$

where $f_{(i)} := f(\mathcal{A}, (S \cup \widetilde{S})_{(i)})$. To relate empirical leave-one-out error and population error with hypothesis stability condition, we use the following lemma:

Lemma G.2.2 (Bousquet and Elisseeff (2002))**.** *For the leave-one-out error, we have*

$$\mathbb{E}\left[\left(\mathcal{E}_{\mathcal{D}'}(\widehat{w}f) - \mathcal{E}_{LOO(\widetilde{S}_M)}\right)^2\right] \leqslant \frac{1}{2m_1} + \frac{3\beta}{n+m} \,. \tag{G.53}$$

Proof of the above lemma is similar to the proof of Lemma 9 in Bousquet and Elisseeff (2002) and can be found in App. G.4. Before presenting the proof of Theorem 8.4.2, we introduce some more notation. Let $\boldsymbol{X}_{(i)}$ denote the matrix of covariates with the i^{th} point removed. Similarly, let $\boldsymbol{y}_{(i)}$ be the array of responses with the i^{th} point removed. Define the corresponding regularized GD solution as $\widehat{w}w_{(i)} = \left(\boldsymbol{X}_{(i)}^T\boldsymbol{X}_{(i)} + \lambda\boldsymbol{I}\right)^{-1}\boldsymbol{X}_{(i)}^T\boldsymbol{y}_{(i)}$. Define $\widehat{w}f_{(i)}(x) := f(x; \widehat{w}w_{(i)})$.

Proof of Theorem 8.4.2. Because squared loss minimization does not imply 0-1 error minimization, we cannot use arguments from Lemma 8.3.2. This is the main technical difficulty. To compare the 0-1 error at a train point with an unseen point, we use the closed-form expression for $\widehat{w}$ and Shermann-Morrison formula to upper bound training error with leave-one-out cross validation error.

The proof is divided into three parts: In part one, we show that 0-1 error on mislabeled points in the training set is lower than the error obtained by leave-one-out error at those points. In part two, we relate this leave-one-out error with the population error on mislabeled distribution using Condition 8.4.1. While the empirical leave-one-out error is an unbiased estimator of the average population error of leave-one-out classifiers, we need hypothesis stability to control the variance of empirical leave-one-out error. Finally, in part three, we show that the error on the mislabeled training points can be estimated with just the randomly labeled and clean training data (as in proof of Theorem 8.3.1).

Part 1 First we relate training error with leave-one-out error. For any training point (x_i, y_i) in $\widetilde{S} \cup S$, we have

$$\mathcal{E}(\widehat{w}f(x_i), y_i) = \mathbb{I}\left[y_i \cdot x_i^T\widehat{w}w < 0\right] = \mathbb{I}\left[y_i \cdot x_i^T\left(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y} < 0\right] \tag{G.54}$$

$$= \mathbb{I}\left[y_i \cdot x_i^T\underbrace{\left(\boldsymbol{X}_{(i)}^T\boldsymbol{X}_{(i)} + x_i^Tx_i + \lambda\boldsymbol{I}\right)^{-1}}_{\text{I}}(\boldsymbol{X}_{(i)}^T\boldsymbol{y}_{(i)} + y_i \cdot x_i) < 0\right]. \tag{G.55}$$

Letting $\boldsymbol{A} = \left(\boldsymbol{X}_{(i)}^T\boldsymbol{X}_{(i)} + \lambda\boldsymbol{I}\right)$ and using Lemma G.2.1 on term 1, we have

$$\mathcal{E}(\widehat{w}f(x_i), y_i) = \mathbb{I}\left[y_i \cdot x_i^T\left[\boldsymbol{A}^{-1} - \frac{\boldsymbol{A}^{-1}x_ix_i^T\boldsymbol{A}^{-1}}{1 + x_i^T\boldsymbol{A}^{-1}x_i}\right](\boldsymbol{X}_{(i)}^T\boldsymbol{y}_{(i)} + y_i \cdot x_i) < 0\right] \tag{G.56}$$

$$= \mathbb{I}\left[y_i \cdot \left[\frac{x_i^T\boldsymbol{A}^{-1}(1 + x_i^T\boldsymbol{A}^{-1}x_i) - x_i^T\boldsymbol{A}^{-1}x_ix_i^T\boldsymbol{A}^{-1}}{1 + x_i^T\boldsymbol{A}^{-1}x_i}\right](\boldsymbol{X}_{(i)}^T\boldsymbol{y}_{(i)} + y_i \cdot x_i) < 0\right] \tag{G.57}$$

$$= \mathbb{I}\left[y_i \cdot \left[\frac{x_i^T\boldsymbol{A}^{-1}}{1 + x_i^T\boldsymbol{A}^{-1}x_i}\right](\boldsymbol{X}_{(i)}^T\boldsymbol{y}_{(i)} + y_i \cdot x_i) < 0\right]. \tag{G.58}$$

Since $1 + x_i^T\boldsymbol{A}^{-1}x_i > 0$, we have

$$\mathcal{E}(\widehat{w}f(x_i), y_i) = \mathbb{I}\left[y_i \cdot x_i^T\boldsymbol{A}^{-1}(\boldsymbol{X}_{(i)}^T\boldsymbol{y}_{(i)} + y_i \cdot x_i) < 0\right] \tag{G.59}$$

$$= \mathbb{I}\left[x_i^T\boldsymbol{A}^{-1}x_i + y_i \cdot x_i^T\boldsymbol{A}^{-1}(\boldsymbol{X}_{(i)}^T\boldsymbol{y}_{(i)}) < 0\right] \tag{G.60}$$

$$\leqslant \mathbb{I}\left[y_i \cdot x_i^T \boldsymbol{A}^{-1}(\boldsymbol{X}_{(i)}^T \boldsymbol{y}_{(i)}) < 0\right] = \mathcal{E}(\widehat{w} f_{(i)}(x_i), y_i)\,. \tag{G.61}$$

Using (G.61), we have

$$\mathcal{E}_{\widetilde{S}_M}(\widehat{w} f) \leqslant \mathcal{E}_{\mathrm{LOO}(\widetilde{S}_M)} := \frac{\sum_{(x_i,y_i)\in\widetilde{S}_M} \mathcal{E}(\widehat{w} f_{(i)}(x_i), y_i)}{\left|\widetilde{S}_M\right|}\,. \tag{G.62}$$

Part 2 We now relate RHS in (G.62) with the population error on mislabeled distribution. To do this, we leverage Condition 8.4.1 and Lemma G.2.2. In particular, we have

$$\mathbb{E}_{\mathcal{S}\cup\widetilde{S}_M}\left[\left(\mathcal{E}_{\mathcal{D}'}(\widehat{w} f) - \mathcal{E}_{\mathrm{LOO}(\widetilde{S}_M)}\right)^2\right] \leqslant \frac{1}{2m_1} + \frac{3\beta}{m+n}\,. \tag{G.63}$$

Using Chebyshev's inequality, with probability at least $1-\delta$, we have

$$\mathcal{E}_{\mathrm{LOO}(\widetilde{S}_M)} \leqslant \mathcal{E}_{\mathcal{D}'}(\widehat{w} f) + \sqrt{\frac{1}{\delta}\left(\frac{1}{2m_1} + \frac{3\beta}{m+n}\right)}\,. \tag{G.64}$$

Part 3 Combining (G.64) and (G.62), we have

$$\mathcal{E}_{\widetilde{S}_M}(\widehat{w} f) \leqslant \mathcal{E}_{\mathcal{D}'}(\widehat{w} f) + \sqrt{\frac{1}{\delta}\left(\frac{1}{2m_1} + \frac{3\beta}{m+n}\right)}\,. \tag{G.65}$$

Compare (G.65) with (G.8) in the proof of Lemma 8.3.2. We obtain a similar relationship between $\mathcal{E}_{\widetilde{S}_M}$ and $\mathcal{E}_{\mathcal{D}'}$ but with a polynomial concentration instead of exponential concentration. In addition, since we just use concentration arguments to relate mislabeled error to the errors on the clean and unlabeled portions of the randomly labeled data, we can directly use the results in Lemma 8.3.3 and Lemma 8.3.4. Therefore, combining results in Lemma 8.3.3, Lemma 8.3.4, and (G.65) with union bound, we have with probability at least $1-\delta$

$$\mathcal{E}_{\mathcal{D}}(\widehat{f}) \leqslant \mathcal{E}_{\mathcal{S}}(\widehat{f}) + 1 - 2\mathcal{E}_{\widetilde{\mathcal{S}}}(\widehat{f}) + \left(\sqrt{2}\mathcal{E}_{\widetilde{\mathcal{S}}}(\widehat{f}) + 1 + \frac{m}{2n}\right)\sqrt{\frac{\log(4/\delta)}{m}} + \sqrt{\frac{4}{\delta}\left(\frac{1}{m} + \frac{3\beta}{m+n}\right)}\,. \tag{G.66}$$

□

G.2.2 Extension to multiclass classification

For multiclass problems with squared loss minimization, as standard practice, we consider one-hot encoding for the underlying label, i.e., a class label $c \in [k]$ is treated as $(0, \cdot, 0, 1, 0, \cdot, 0) \in \mathbb{R}^k$ (with c-th coordinate being 1). As before, we suppose that the parameters of the linear function are obtained via gradient descent on the following L_2 regularized problem:

$$\mathcal{L}_S(w; \lambda) := \sum_{i=1}^{n} \left\| w^T x_i - y_i \right\| 2^2 + \lambda \sum_{j=1}^{k} \left\| w_j \right\| 2^2 \,, \tag{G.67}$$

where $\lambda \geqslant 0$ is a regularization parameter. We assume access to a clean dataset $S = \{(x_i, y_i)\}_{i=1}^n \sim \mathcal{D}^n$ and randomly labeled dataset $\widetilde{S} = \{(x_i, y_i)\}_{i=n+1}^{n+m} \sim \widetilde{\mathcal{D}}^m$. Let $\boldsymbol{X} = [x_1, x_2, \cdots, x_{m+n}]$ and $\boldsymbol{y} = [e_{y_1}, e_{y_2}, \cdots, e_{y_{m+n}}]$. Fix a positive learning rate η such that $\eta \leqslant 1/\left(\left\|\boldsymbol{X}^T\boldsymbol{X}\right\| \text{op} + \lambda^2\right)$ and an initialization $w_0 = 0$. Consider the following gradient descent iterates to minimize objective (G.50) on $S \cup \widetilde{S}$:

$$w_j{}^t = w_j{}^{t-1} - \eta \nabla_{w_j} \mathcal{L}_{S \cup \widetilde{S}}(w^{t-1}; \lambda) \quad \forall t = 1, 2, \ldots \text{ and } j = 1, 2, \ldots, k \,. \tag{G.68}$$

Then we have $\{w_j{}^t\}$ for all $j = 1, 2, \cdots, k$ converge to the limiting solution $\widehat{w}w_j = \left(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I}\right)^{-1} \boldsymbol{X}^T\boldsymbol{y}_j$. Define $\widehat{f}(x) := f(x; \widehat{w}w)$.

Theorem G.2.3. *Assume that this gradient descent algorithm satisfies Condition 8.4.1 with* $\beta = \mathcal{O}(1)$. *Then for a multiclass classification problem wth* k *classes, for any* $\delta > 0$, *with probability at least* $1 - \delta$, *we have:*

$$\begin{aligned}\mathcal{E}_{\mathcal{D}}(\widehat{f}) \leqslant \mathcal{E}_{\mathcal{S}}(\widehat{f}) + (k-1)\left(1 - \frac{k}{k-1}\mathcal{E}_{\widetilde{\mathcal{S}}}(\widehat{f})\right) \\ + \left(k + \sqrt{k} + \frac{m}{n\sqrt{k}}\right)\sqrt{\frac{\log(4/\delta)}{2m}} + \sqrt{k(k-1)}\sqrt{\frac{4}{\delta}\left(\frac{1}{m} + \frac{3\beta}{m+n}\right)} \,.\end{aligned} \tag{G.69}$$

Proof. The proof of this theorem is divided into two parts. In the first part, we relate the error on the mislabeled samples with the population error on the mislabeled data. Similar to the proof of Theorem 8.4.2, we use Shermann-Morrison formula to upper bound training error with leave-one-out error on each $\widehat{w}w^j$. Second part of the proof follows entirely from the proof of Theorem 8.3.7. In essence, the first part derives an equivalent of (G.36) for GD training with squared loss and then the second part follows from the proof of Theorem 8.3.7.

Part-1: Consider a training point (x_i, y_i) in $\widetilde{S} \cup S$. For simplicity, we use c_i to denote the class of i-th point and use y_i as the corresponding one-hot embedding. Recall error in multiclass point is given by $\mathcal{E}(\widehat{w}f(x_i), y_i) = \mathbb{I}\left[c_i \notin \arg\max x_i^T \widehat{w}w\right]$. Thus, there exists a $j \neq c_i \in [k]$, such that we have

$$\begin{aligned}\mathcal{E}(\widehat{w}f(x_i), y_i) &= \mathbb{I}\left[c_i \notin \arg\max x_i^T \widehat{w}w\right] = \mathbb{I}\left[x_i^T \widehat{w}w_{c_i} < x_i^T \widehat{w}w_j\right] && \text{(G.70)} \\ &= \mathbb{I}\left[x_i^T \left(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I}\right)^{-1} \boldsymbol{X}^T\boldsymbol{y}_{c_i} < x_i^T \left(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I}\right)^{-1} \boldsymbol{X}^T\boldsymbol{y}_j\right] && \text{(G.71)}\end{aligned}$$

$$= \mathbb{I}\left[x_i^T \underbrace{\left(\boldsymbol{X}_{(i)}^T \boldsymbol{X}_{(i)} + x_i^T x_i + \lambda \boldsymbol{I}\right)^{-1}}_{\text{I}} \left(\boldsymbol{X}_{(i)}^T \boldsymbol{y}_{c_i\,(i)} + x_i - \boldsymbol{X}_{(i)}^T \boldsymbol{y}_{j\,(i)}\right) < 0 \right]. \tag{G.72}$$

Letting $\boldsymbol{A} = \left(\boldsymbol{X}_{(i)}^T \boldsymbol{X}_{(i)} + \lambda \boldsymbol{I}\right)$ and using Lemma G.2.1 on term 1, we have

$$\mathcal{E}(\widehat{w}f(x_i), y_i) = \mathbb{I}\left[x_i^T \left[\boldsymbol{A}^{-1} - \frac{\boldsymbol{A}^{-1} x_i x_i^T \boldsymbol{A}^{-1}}{1 + x_i^T \boldsymbol{A}^{-1} x_i} \right] \left(\boldsymbol{X}_{(i)}^T \boldsymbol{y}_{c_i\,(i)} + x_i - \boldsymbol{X}_{(i)}^T \boldsymbol{y}_{j\,(i)}\right) < 0 \right] \tag{G.73}$$

$$= \mathbb{I}\left[\left[\frac{x_i^T \boldsymbol{A}^{-1}(1 + x_i^T \boldsymbol{A}^{-1} x_i) - x_i^T \boldsymbol{A}^{-1} x_i x_i^T \boldsymbol{A}^{-1}}{1 + x_i^T \boldsymbol{A}^{-1} x_i} \right] \left(\boldsymbol{X}_{(i)}^T \boldsymbol{y}_{c_i\,(i)} + x_i - \boldsymbol{X}_{(i)}^T \boldsymbol{y}_{j\,(i)}\right) < 0 \right] \tag{G.74}$$

$$= \mathbb{I}\left[\left[\frac{x_i^T \boldsymbol{A}^{-1}}{1 + x_i^T \boldsymbol{A}^{-1} x_i} \right] \left(\boldsymbol{X}_{(i)}^T \boldsymbol{y}_{c_i\,(i)} + x_i - \boldsymbol{X}_{(i)}^T \boldsymbol{y}_{j\,(i)}\right) < 0 \right]. \tag{G.75}$$

Since $1 + x_i^T \boldsymbol{A}^{-1} x_i > 0$, we have

$$\mathcal{E}(\widehat{w}f(x_i), y_i) = \mathbb{I}\left[x_i^T \boldsymbol{A}^{-1} \left(\boldsymbol{X}_{(i)}^T \boldsymbol{y}_{c_i\,(i)} + x_i - \boldsymbol{X}_{(i)}^T \boldsymbol{y}_{j\,(i)}\right) < 0 \right] \tag{G.76}$$

$$= \mathbb{I}\left[x_i^T \boldsymbol{A}^{-1} x_i + x_i^T \boldsymbol{A}^{-1} \boldsymbol{X}_{(i)}^T \boldsymbol{y}_{c_i\,(i)} - x_i^T \boldsymbol{A}^{-1} \boldsymbol{X}_{(i)}^T \boldsymbol{y}_{j\,(i)} < 0 \right] \tag{G.77}$$

$$\leqslant \mathbb{I}\left[x_i^T \boldsymbol{A}^{-1} \boldsymbol{X}_{(i)}^T \boldsymbol{y}_{c_i\,(i)} - x_i^T \boldsymbol{A}^{-1} \boldsymbol{X}_{(i)}^T \boldsymbol{y}_{j\,(i)} < 0 \right] = \mathcal{E}(\widehat{w}f_{(i)}(x_i), y_i). \tag{G.78}$$

Using (G.78), we have

$$\mathcal{E}_{\widetilde{S}_M}(\widehat{w}f) \leqslant \mathcal{E}_{\mathrm{LOO}(\widetilde{S}_M)} := \frac{\sum_{(x_i, y_i) \in \widetilde{S}_M} \mathcal{E}(\widehat{w}f_{(i)}(x_i), y_i)}{\left|\widetilde{S}_M\right|}. \tag{G.79}$$

We now relate RHS in (G.62) with the population error on mislabeled distribution. Similar as before, to do this, we leverage Condition 8.4.1 and Lemma G.2.2. Using (G.64) and (G.79), we have

$$\mathcal{E}_{\widetilde{S}_M}(\widehat{w}f) \leqslant \mathcal{E}_{\mathcal{D}'}(\widehat{w}f) + \sqrt{\frac{1}{\delta}\left(\frac{1}{2m_1} + \frac{3\beta}{m+n}\right)}. \tag{G.80}$$

We have now derived a parallel to (G.36). Using the same arguments in the proof of Lemma G.1.2, we have

$$\mathcal{E}_{\mathcal{D}}(\widehat{w}f) \leqslant (k-1)\left(1 - \mathcal{E}_{\widetilde{S}_M}(\widehat{w}f)\right) + (k-1)\sqrt{\frac{k}{\delta(k-1)}\left(\frac{1}{2m_1} + \frac{3\beta}{m+n}\right)}. \tag{G.81}$$

Part-2: We now combine the results in Lemma G.1.3 and Lemma G.1.4 to obtain the final inequality in terms of quantities that can be computed from just the randomly labeled and clean data. Similar to the binary case, we obtained a polynomial concentration instead of exponential concentration. Combining (G.81) with Lemma G.1.3 and Lemma G.1.4, we have with probability at least $1 - \delta$

$$\begin{aligned}\mathcal{E}_{\mathcal{D}}(\widehat{f}) \leqslant \mathcal{E}_{\mathcal{S}}(\widehat{f}) + (k-1)\left(1 - \frac{k}{k-1}\mathcal{E}_{\widetilde{\mathcal{S}}}(\widehat{f})\right) \\ + \left(k + \sqrt{k} + \frac{m}{n\sqrt{k}}\right)\sqrt{\frac{\log(4/\delta)}{2m}} + \sqrt{k(k-1)}\sqrt{\frac{4}{\delta}\left(\frac{1}{m} + \frac{3\beta}{m+n}\right)}\,. \end{aligned} \tag{G.82}$$

□

G.2.3 Discussion on Condition 8.4.1

The quantity in LHS of Condition 8.4.1 measures how much the function learned by the algorithm (in terms of error on unseen point) will change when one point in the training set is removed. We need hypothesis stability condition to control the variance of the empirical leave-one-out error to show concentration of average leave-one-error with the population error.

Additionally, we note that while the dominating term in the RHS of Theorem 8.4.2 matches with the dominating term in ERM bound in Theorem 8.3.1, there is a polynomial concentration term (dependence on $1/\delta$ instead of $\log(\sqrt{1/\delta})$) in Theorem 8.4.2. Since with hypothesis stability, we just bound the variance, the polynomial concentration is due to the use of Chebyshev's inequality instead of an exponential tail inequality (as in Lemma 8.3.2). Recent works have highlighted that a slightly stronger condition than hypothesis stability can be used to obtain an exponential concentration for leave-one-out error (Abou-Moustafa and Szepesvári, 2019), but we leave this for future work for now.

G.2.4 Formal statement and proof of Proposition 8.4.3

Before formally presenting the result, we will introduce some notation. By $\mathcal{L}_S(w)$, we denote the objective in (G.50) with $\lambda = 0$. Assume Singular Value Decomposition (SVD) of $\boldsymbol{X}$ as $\sqrt{n}\boldsymbol{U}\boldsymbol{S}^{1/2}\boldsymbol{V}^T$. Hence $\boldsymbol{X}^T\boldsymbol{X} = \boldsymbol{V}\boldsymbol{S}\boldsymbol{V}^T$. Consider the GD iterates defined in (G.51). We now derive closed form expression for the t^{th} iterate of gradient descent:

$$w_t = w_{t-1} + \eta \cdot \boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X}w_{t-1}) = (\boldsymbol{I} - \eta\boldsymbol{V}\boldsymbol{S}\boldsymbol{V}^T)w_{k-1} + \eta\boldsymbol{X}^T\boldsymbol{y}\,. \tag{G.83}$$

Rotating by $\boldsymbol{V}^T$, we get

$$\widetilde{w}_t = (\boldsymbol{I} - \eta\boldsymbol{S})\widetilde{w}_{k-1} + \eta\widetilde{\boldsymbol{y}}, \tag{G.84}$$

where $\widetilde{w}_t = \boldsymbol{V}^T w_t$ and $\widetilde{\boldsymbol{y}} = \boldsymbol{V}^T\boldsymbol{X}^T\boldsymbol{y}$. Assuming the initial point $w_0 = 0$ and applying the recursion in (G.84), we get

$$\widetilde{w}_t = \boldsymbol{S}^{-1}(\boldsymbol{I} - (\boldsymbol{I} - \eta\boldsymbol{S})^k)\widetilde{\boldsymbol{y}}\,, \tag{G.85}$$

Projecting solution back to the original space, we have

$$w_t = \boldsymbol{V}\boldsymbol{S}^{-1}(\boldsymbol{I} - (\boldsymbol{I} - \eta\boldsymbol{S})^k)\boldsymbol{V}^T\boldsymbol{X}^T\boldsymbol{y}\,. \tag{G.86}$$

Define $f_t(x) := f(x; w_t)$ as the solution at the t^{th} iterate. Let $\widetilde{w}_\lambda = \arg\min_w \mathcal{L}_{\mathcal{S}}(w; \lambda) = (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^T\boldsymbol{y} = \boldsymbol{V}(\boldsymbol{S} + \lambda\boldsymbol{I})^{-1}\boldsymbol{V}^T\boldsymbol{X}^T\boldsymbol{y}$. and define $\widetilde{f}_\lambda(x) := f(x; \widetilde{w}_\lambda)$ as the regularized solution. Assume κ be the condition number of the population covariance matrix and let $s_{\min}$ be the minimum positive singular value of the empirical covariance matrix. Our proof idea is inspired from recent work on relating gradient flow solution and regularized solution for regression problems (Ali et al., 2018). We will use the following lemma in the proof:

Lemma G.2.4. *For all $x \in [0, 1]$ and for all $k \in \mathbb{N}$, we have (a) $\frac{kx}{1+kx} \leqslant 1 - (1-x)^k$ and (b) $1 - (1-x)^k \leqslant 2 \cdot \frac{kx}{kx+1}$.*

Proof. Using $(1-x)^k \leqslant \frac{1}{1+kx}$, we have part (a). For part (b), we numerically maximize $\frac{(1+kx)(1-(1-x)^k)}{kx}$ for all $k \geqslant 1$ and for all $x \in [0, 1]$. □

Proposition G.2.5 (Formal statement of Proposition 8.4.3)**.** *Let $\lambda = \frac{1}{t\eta}$. For a training point x, we have*

$$\mathbb{E}_{x\sim\mathcal{S}}\left[(f_t(x) - \widetilde{f}_\lambda(x))^2\right] \leqslant c(t,\eta) \cdot \mathbb{E}_{x\sim\mathcal{S}}\left[f_t(x)^2\right]\,,$$

where $c(t, \eta) := \min(0.25, \frac{1}{s^2_{min}t^2\eta^2})$. Similarly for a test point, we have

$$\mathbb{E}_{x\sim\mathcal{D}_{\mathcal{X}}}\left[(f_t(x) - \widetilde{f}_\lambda(x))^2\right] \leqslant \kappa \cdot c(t,\eta) \cdot \mathbb{E}_{x\sim\mathcal{D}_{\mathcal{X}}}\left[f_t(x)^2\right]\,.$$

Proof. We want to analyze the expected squared difference output of regularized linear regression with regularization constant $\lambda = \frac{1}{\eta t}$ and the gradient descent solution at the t^{th} iterate. We separately expand the algebraic expression for squared difference at a training point and a test point. Then the main step is to show that $\left[\boldsymbol{S}^{-1}(\boldsymbol{I} - (\boldsymbol{I} - \eta\boldsymbol{S})^k) - (\boldsymbol{S} + \lambda\boldsymbol{I})^{-1}\right] \preceq c(\eta, t) \cdot \boldsymbol{S}^{-1}(\boldsymbol{I} - (\boldsymbol{I} - \eta\boldsymbol{S})^k)$.

Part 1 First, we will analyze the squared difference of the output at a training point (for simplicity, we refer to $S \cup \widetilde{S}$ as S), i.e.,

$$\mathbb{E}_{x\sim\mathcal{S}}\left[\left(f_t(x) - \widetilde{f}_\lambda(x)\right)^2\right] = \left\|\boldsymbol{X}w_t - \boldsymbol{X}\widetilde{w}_\lambda\right\| 2^2 \tag{G.87}$$

$$= \left\|\boldsymbol{X}\boldsymbol{V}\boldsymbol{S}^{-1}(\boldsymbol{I} - (\boldsymbol{I} - \eta\boldsymbol{S})^t)\boldsymbol{V}^T\boldsymbol{X}^T\boldsymbol{y} - \boldsymbol{X}\boldsymbol{V}(\boldsymbol{S} + \lambda\boldsymbol{I})^{-1}\boldsymbol{V}^T\boldsymbol{X}^T\boldsymbol{y}\right\| 2^2 \tag{G.88}$$

$$= \left\|\boldsymbol{X}\boldsymbol{V}\left(\boldsymbol{S}^{-1}(\boldsymbol{I} - (\boldsymbol{I} - \eta\boldsymbol{S})^t) - (\boldsymbol{S} + \lambda\boldsymbol{I})^{-1}\right)\boldsymbol{V}^T\boldsymbol{X}^T\boldsymbol{y}\right\| 2 \tag{G.89}$$

$$= \boldsymbol{y}^T\boldsymbol{V}\boldsymbol{X}\left(\underbrace{\boldsymbol{S}^{-1}(\boldsymbol{I} - (\boldsymbol{I} - \eta\boldsymbol{S})^t) - (\boldsymbol{S} + \lambda\boldsymbol{I})^{-1}}_{\text{I}}\right)^2\boldsymbol{S}\boldsymbol{V}^T\boldsymbol{X}^T\boldsymbol{y}\,. \tag{G.90}$$

We now separately consider term 1. Substituting $\lambda = \frac{1}{t\eta}$, we get

$$\begin{aligned}
\boldsymbol{S}^{-1}(\boldsymbol{I} - (\boldsymbol{I} - \eta\boldsymbol{S})^t) - (\boldsymbol{S} + \lambda\boldsymbol{I})^{-1} &= \boldsymbol{S}^{-1}\left((\boldsymbol{I} - (\boldsymbol{I} - \eta\boldsymbol{S})^t) - (\boldsymbol{I} + \boldsymbol{S}^{-1}\lambda)^{-1}\right) && \text{(G.91)}\\
&= \underbrace{\boldsymbol{S}^{-1}\left((\boldsymbol{I} - (\boldsymbol{I} - \eta\boldsymbol{S})^t) - (\boldsymbol{I} + (\boldsymbol{S}t\eta)^{-1})^{-1}\right)}_{\boldsymbol{A}}\,. && \text{(G.92)}
\end{aligned}$$

We now separately bound the diagonal entries in matrix $\boldsymbol{A}$. With s_i, we denote i^{th} diagonal entry of $\boldsymbol{S}$. Note that since $\eta \leqslant 1/\left\|S\right\|\text{op}$, for all i, $\eta s_i \leqslant 1$. Consider i^{th} diagonal term (which is non-zero) of the diagonal matrix $\boldsymbol{A}$, we have

$$\begin{aligned}
\boldsymbol{A}_{ii} = \frac{1}{s_i}\left(1 - (1 - s_i\eta)^t - \frac{t\eta s_i}{1 + t\eta s_i}\right) &= \frac{1 - (1 - s_i\eta)^t}{s_i}\left(\underbrace{1 - \frac{t\eta s_i}{(1 + t\eta s_i)(1 - (1 - s_i\eta)^t)}}_{\text{II}}\right) && \text{(G.93)}\\
&\leqslant \frac{1}{2}\left[\frac{1 - (1 - s_i\eta)^t}{s_i}\right]. && \text{(Using Lemma G.2.4 (b))}
\end{aligned}$$

Additionally, we can also show the following upper bound on term 2:

$$\begin{aligned}
1 - \frac{t\eta s_i}{(1 + t\eta s_i)(1 - (1 - s_i\eta)^t)} &= \frac{(1 + t\eta s_i)(1 - (1 - s_i\eta)^t) - t\eta s_i}{(1 + t\eta s_i)(1 - (1 - s_i\eta)^t)} && \text{(G.94)}\\
&\leqslant \frac{1 - (1 - s_i\eta)^t - t\eta s_i(1 - s_i\eta)^t}{(1 + t\eta s_i)(1 - (1 - s_i\eta)^t)} && \text{(G.95)}\\
&\leqslant \frac{1}{t\eta s_i}\,. && \text{(Using Lemma G.2.4 (a))}
\end{aligned}$$

Combining both the upper bounds on each diagonal entry $\boldsymbol{A}_{ii}$, we have

$$\boldsymbol{A} \preceq c_1(\eta, t) \cdot \boldsymbol{S}^{-1}(\boldsymbol{I} - (\boldsymbol{I} - \eta\boldsymbol{S})^t)\,, \tag{G.96}$$

where $c_1(\eta, t) = \min(0.5, \frac{1}{ts_i\eta})$. Plugging this into (G.90), we have

$$\begin{aligned}
\mathbb{E}_{x\sim\mathcal{S}}\left[\left(f_t(x) - \widetilde{f}_\lambda(x)\right)^2\right] &\leqslant c(\eta, t) \cdot \boldsymbol{y}^T\boldsymbol{V}\boldsymbol{X}\left(\boldsymbol{S}^{-1}(\boldsymbol{I} - (\boldsymbol{I} - \eta\boldsymbol{S})^t)\right)^2\boldsymbol{S}\boldsymbol{V}^T\boldsymbol{X}^T\boldsymbol{y} && \text{(G.97)}\\
&= c(\eta, t) \cdot \boldsymbol{y}^T\boldsymbol{V}\boldsymbol{X}\left(\boldsymbol{S}^{-1}(\boldsymbol{I} - (\boldsymbol{I} - \eta\boldsymbol{S})^t)\right)\boldsymbol{S}\left(\boldsymbol{S}^{-1}(\boldsymbol{I} - (\boldsymbol{I} - \eta\boldsymbol{S})^t)\right)\boldsymbol{V}^T\boldsymbol{X}^T\boldsymbol{y} && \text{(G.98)}\\
&= c(\eta, t) \cdot \left\|\boldsymbol{X}w_t\right\| 2^2 && \text{(G.99)}\\
&= c(\eta, t) \cdot \mathbb{E}_{x\sim\mathcal{S}}\left[(f_t(x))^2\right], && \text{(G.100)}
\end{aligned}$$

where $c(\eta, t) = \min(0.25, \frac{1}{t^2 s_i^2 \eta^2})$.

Part 2 With $\boldsymbol{\Sigma}$, we denote the underlying true covariance matrix. We now consider the squared difference of output at an unseen point:

$$\mathbb{E}_{x\sim\mathcal{D}_\mathcal{X}}\left[\left(f_t(x)-\widetilde{f}_\lambda(x)\right)^2\right] = \mathbb{E}_{x\sim\mathcal{D}_\mathcal{X}}\left[\left\|x^T w_t - x^T \widetilde{w}_\lambda\right\| 2\right] \tag{G.101}$$

$$= \left\|x^T \boldsymbol{V}\boldsymbol{S}^{-1}(\boldsymbol{I}-(\boldsymbol{I}-\eta\boldsymbol{S})^t)\boldsymbol{V}^T\boldsymbol{X}^T\boldsymbol{y} - x^T\boldsymbol{V}(\boldsymbol{S}+\lambda\boldsymbol{I})^{-1}\boldsymbol{V}^T\boldsymbol{X}^T\boldsymbol{y}\right\| 2 \tag{G.102}$$

$$= \left\|x^T \boldsymbol{V}\left(\boldsymbol{S}^{-1}(\boldsymbol{I}-(\boldsymbol{I}-\eta\boldsymbol{S})^t) - (\boldsymbol{S}+\lambda\boldsymbol{I})^{-1}\right)\boldsymbol{V}^T\boldsymbol{X}^T\boldsymbol{y}\right\| 2 \tag{G.103}$$

$$= \boldsymbol{y}^T\boldsymbol{V}\boldsymbol{X}\left(\boldsymbol{S}^{-1}(\boldsymbol{I}-(\boldsymbol{I}-\eta\boldsymbol{S})^t) - (\boldsymbol{S}+\lambda\boldsymbol{I})^{-1}\right)\boldsymbol{V}^T\boldsymbol{\Sigma}\boldsymbol{V} \tag{G.104}$$

$$\left((\boldsymbol{I}-(\boldsymbol{I}-\eta\boldsymbol{S})^t) - (\boldsymbol{S}+\lambda\boldsymbol{I})^{-1}\right)\boldsymbol{V}^T\boldsymbol{X}^T\boldsymbol{y} \tag{G.105}$$

$$\leqslant \sigma_{\max}\cdot\boldsymbol{y}^T\boldsymbol{V}\boldsymbol{X}\left(\underbrace{\boldsymbol{S}^{-1}(\boldsymbol{I}-(\boldsymbol{I}-\eta\boldsymbol{S})^t) - (\boldsymbol{S}+\lambda\boldsymbol{I})^{-1}}_{\mathrm{I}}\right)^2\boldsymbol{V}^T\boldsymbol{X}^T\boldsymbol{y}\,, \tag{G.106}$$

where $\sigma_{\max}$ is the maximum eigenvalue of the underlying covariance matrix $\boldsymbol{\Sigma}$. Using the upper bound on term 1 in (G.96), we have

$$\mathbb{E}_{x\sim\mathcal{D}_\mathcal{X}}\left[\left(f_t(x)-\widetilde{f}_\lambda(x)\right)^2\right] \leqslant \sigma_{\max}\cdot c(\eta,t)\cdot\boldsymbol{y}^T\boldsymbol{V}\boldsymbol{X}\left(\boldsymbol{S}^{-1}(\boldsymbol{I}-(\boldsymbol{I}-\eta\boldsymbol{S})^t)\right)^2\boldsymbol{V}^T\boldsymbol{X}^T\boldsymbol{y} \tag{G.107}$$

$$= \kappa\cdot c(\eta,t)\cdot\sigma_{\min}\cdot\left\|\boldsymbol{V}\left(\boldsymbol{S}^{-1}(\boldsymbol{I}-(\boldsymbol{I}-\eta\boldsymbol{S})^t)\right)\boldsymbol{V}^T\boldsymbol{X}^T\boldsymbol{y}\right\| 2^2 \tag{G.108}$$

$$\leqslant \kappa\cdot c(\eta,t)\cdot\left[\boldsymbol{V}\left(\boldsymbol{S}^{-1}(\boldsymbol{I}-(\boldsymbol{I}-\eta\boldsymbol{S})^t)\right)\boldsymbol{V}^T\boldsymbol{X}^T\right]^T\boldsymbol{\Sigma} \tag{G.109}$$

$$\left[\boldsymbol{V}\left(\boldsymbol{S}^{-1}(\boldsymbol{I}-(\boldsymbol{I}-\eta\boldsymbol{S})^t)\right)\boldsymbol{V}^T\boldsymbol{X}^T\right]\boldsymbol{y} \tag{G.110}$$

$$= \kappa\cdot c(\eta,t)\cdot\mathbb{E}_{x\sim\mathcal{D}_\mathcal{X}}\left[\left\|x^T w_t\right\| 2\right]\,. \tag{G.111}$$

□

G.2.5 Extension to deep learning

Under Assumption G.2.6, we present the formal result parallel to Theorem 8.3.7.

Theorem G.2.6. *Consider a multiclass classification problem with k classes. Under Assumption 2, for any $\delta > 0$, with probability at least $1-\delta$, we have*

$$\mathcal{E}_\mathcal{D}(\widehat{f}) \leqslant \mathcal{E}_\mathcal{S}(\widehat{f}) + (k-1)\left(1 - \tfrac{k}{k-1}\mathcal{E}_{\widetilde{\mathcal{S}}}(\widehat{f})\right) + c\sqrt{\frac{\log(\frac{4}{\delta})}{2m}}\,, \tag{G.112}$$

for some constant $c \leqslant ((c+1)k + \sqrt{k} + \frac{m}{n\sqrt{k}})$.

The proof follows exactly as in step (i) to (iii) in Theorem 8.3.7.

G.2.6 Justifying Assumption 2

Motivated by the analysis on linear models, we now discuss alternate (and weaker) conditions that imply Assumption 2. We need hypothesis stability (Condition 8.4.1) and the following assumption relating training error and leave-one-error:

Assumption 6. *Let* $\widehat{w}f$ *be a model obtained by training with algorithm* $\mathcal{A}$ *on a mixture of clean* S *and randomly labeled data* $\widetilde{S}$. *Then we assume we have*

$$\mathcal{E}_{\widetilde{\mathcal{S}}_M}(\widehat{w}f) \leqslant \mathcal{E}_{LOO(\widetilde{\mathcal{S}}_M)}\,,$$

for all $(x_i, y_i) \in \widetilde{S}_M$ *where* $\widehat{w}f_{(i)} := f(\mathcal{A}, S \cup \widetilde{S}_{M(i)})$ *and* $\mathcal{E}_{LOO(\widetilde{\mathcal{S}}_M)} := \frac{\sum_{(x_i, y_i) \in \widetilde{\mathcal{S}}_M} \mathcal{E}(\widehat{w}f_{(i)}(x_i), y_i)}{\left|\widetilde{\mathcal{S}}_M\right|}$.

Intuitively, this assumption states that the error on a (mislabeled) datum (x, y) included in the training set is less than the error on that datum (x, y) obtained by a model trained on the training set $S - \{(x, y)\}$. We proved this for linear models trained with GD in the proof of Theorem G.2.3. Condition 8.4.1 with $\beta = \mathcal{O}(1)$ and Assumption 6 together with Lemma G.2.2 implies Assumption 2 with a polynomial residual term (instead of logarithmic in $1/\delta$):

$$\mathcal{E}_{\mathcal{S}_M}(\widehat{w}f) \leqslant \mathcal{E}_{\mathcal{D}'}(\widehat{w}f) + \sqrt{\frac{1}{\delta}\left(\frac{1}{m} + \frac{3\beta}{m+n}\right)}\,. \tag{G.113}$$

G.3 Additional experiments and details

G.3.1 Datasets

Toy Dataset Assume fixed constants μ and σ. For a given label y, we simulate features x in our toy classification setup as follows:

$$x := \texttt{concat}\,[x_1, x_2] \quad \text{where} \quad x_1 \sim \mathcal{N}(y \cdot \mu, \sigma^2 I_{d\times d}) \;\text{ and }\; x_1 \sim \mathcal{N}(0, \sigma^2 I_{d\times d})\,.$$

In experiements throughout the paper, we fix dimention $d = 100$, $\mu = 1.0$, and $\sigma = \sqrt{d}$. Intuitively, x_1 carries the information about the underlying label and x_2 is additional noise independent of the underlying label.

CV datasets We use MNIST (LeCun et al., 1998) and CIFAR10 (Krizhevsky and Hinton, 2009). We produce a binary variant from the multiclass classification problem by mapping classes $\{0, 1, 2, 3, 4\}$ to label 1 and $\{5, 6, 7, 8, 9\}$ to label -1. For CIFAR dataset, we also use the standard data augementation of random crop and horizontal flip. PyTorch code is as follows:

```
(transforms.RandomCrop(32, padding=4),
     transforms.RandomHorizontalFlip())
```

NLP dataset We use IMDb Sentiment analysis (Maas et al., 2011) corpus.

G.3.2 Architecture Details

All experiments were run on NVIDIA GeForce RTX 2080 Ti GPUs. We used PyTorch (Paszke et al., 2019) and Keras with Tensorflow (Abadi et al., 2016) backend for experiments.

Linear model For the toy dataset, we simulate a linear model with scalar output and the same number of parameters as the number of dimensions.

Wide nets To simulate the NTK regime, we experiment with 2−layered wide nets. The PyTorch code for 2-layer wide MLP is as follows:

```
 nn.Sequential(
     nn.Flatten(),
     nn.Linear(input_dims, 200000, bias=True),
     nn.ReLU(),
     nn.Linear(200000, 1, bias=True)
     )
```

We experiment both (i) with the second layer fixed at random initialization; (ii) and updating both layers' weights.

Deep nets for CV tasks We consider a 4-layered MLP. The PyTorch code for 4-layer MLP is as follows:

```
nn.Sequential(nn.Flatten(),
    nn.Linear(input_dim, 5000, bias=True),
    nn.ReLU(),
    nn.Linear(5000, 5000, bias=True),
    nn.ReLU(),
    nn.Linear(5000, 5000, bias=True),
    nn.ReLU(),
    nn.Linear(1024, num_label, bias=True)
    )
```

For MNIST, we use 1000 nodes instead of 5000 nodes in the hidden layer. We also experiment with convolutional nets. In particular, we use ResNet18 (He et al., 2016). Implementation adapted from: https://github.com/kuangliu/pytorch-cifar.git.

Deep nets for NLP We use a simple LSTM model with embeddings intialized with ELMo embeddings (Peters et al., 2018). Code adapted from: https://github.com/kamujun/elmo_experiments/blob/master/elmo_experiment/notebooks/elmo_text_classification_on_imdb.ipynb

We also evaluate our bounds with a BERT model. In particular, we fine-tune an off-the-shelf uncased BERT model (Devlin et al., 2019). Code adapted from Hugging Face Transformers (Wolf et al., 2020): https://huggingface.co/transformers/v3.1.0/custom_datasets.html.

G.3.3 Additonal experiments

Results with SGD on underparameterized linear models

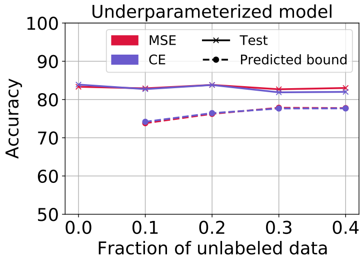
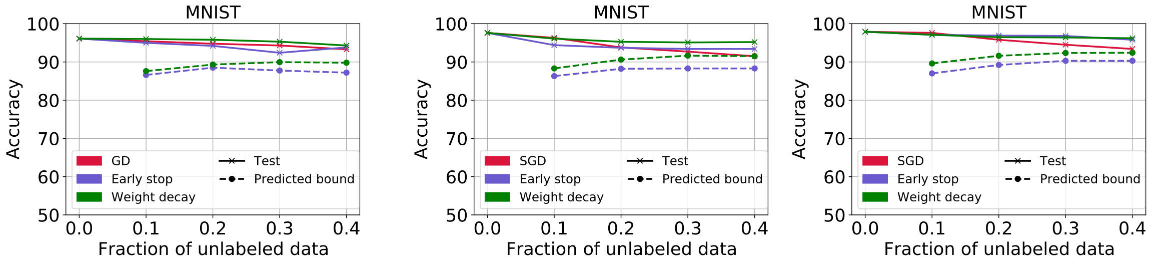


Figure G.1: We plot the accuracy and corresponding bound (RHS in (8.1)) at $\delta = 0.1$ for toy binary classification task. Results aggregated over 3 seeds. Accuracy vs fraction of unlabeled data (w.r.t clean data) in the toy setup with a linear model trained with SGD. Results parallel to Fig. 8.2(a) with SGD.

Results with wide nets on binary MNIST

Results on CIFAR 10 and MNIST We plot epoch wise error curve for results in Table 8.1(Fig. G.3 and Fig. G.4). We observe the same trend as in Fig. 8.1. Additionally, we plot an *oracle bound* obtained by tracking the error on mislabeled data which nevertheless were predicted as true label. To obtain an exact emprical value of the oracle bound, we



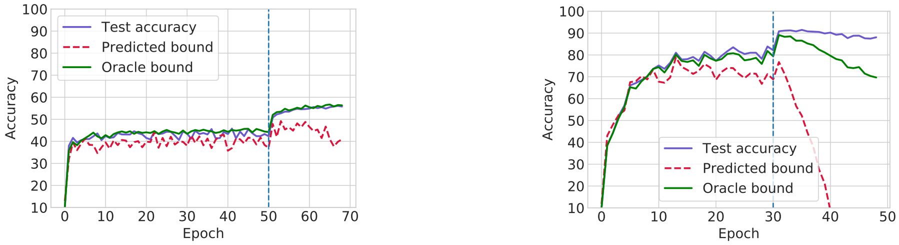
(a) GD with MSE loss

(b) SGD with CE loss

(c) SGD with MSE loss

Figure G.2: We plot the accuracy and corresponding bound (RHS in (8.1)) at $\delta = 0.1$ for binary MNIST classification. Results aggregated over 3 seeds. Accuracy vs fraction of unlabeled data for a 2-layer wide network on binary MNIST with both the layers training in (a,b) and only first layer training in (c). Results parallel to Fig. 8.2(b) .

need underlying true labels for the randomly labeled data. While with just access to extra unlabeled data we cannot calculate oracle bound, we note that the oracle bound is very tight and never violated in practice underscoring an importamt aspect of generalization in multiclass problems. This highlight that even a stronger conjecture may hold in multiclass classification, i.e., error on mislabeled data (where nevertheless true label was predicted) lower bounds the population error on the distribution of mislabeled data and hence, the error on (a specific) mislabeled portion predicts the population accuracy on clean data. On the other hand, the dominating term of in Theorem 8.3.7 is loose when compared with the oracle bound. The main reason, we believe is the pessimistic upper bound in (G.36) in the proof of Lemma G.1.2. We leave an investigation on this gap for future.

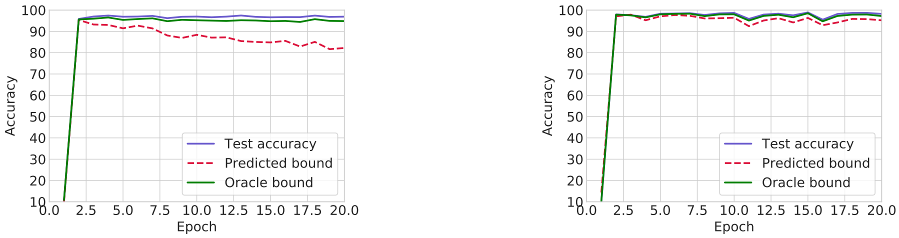


(a) MLP

(b) ResNet

Figure G.3: Per epoch curves for CIFAR10 corresponding results in Table 8.1. As before, we just plot the dominating term in the RHS of (8.5) as predicted bound. Additionally, we also plot the predicted lower bound by the error on mislabeled data which nevertheless were predicted as true label. We refer to this as "Oracle bound". See text for more details.

Results on CIFAR 100 On CIFAR100, our bound in (8.5) yields vacous bounds. However, the oracle bound as explained above yields tight guarantees in the initial phase of the learning (i.e., when learning rate is less than 0.1) (Fig. G.5).



(a) MLP

(b) ResNet

Figure G.4: Per epoch curves for MNIST corresponding results in Table 8.1. As before, we just plot the dominating term in the RHS of (8.5) as predicted bound. Additionally, we also plot the predicted lower bound by the error on mislabeled data which nevertheless were predicted as true label. We refer to this as “Oracle bound”. See text for more details.

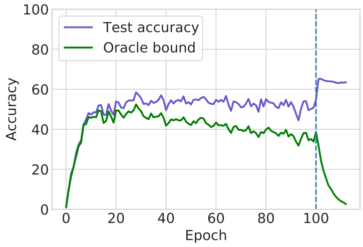


Figure G.5: Predicted lower bound by the error on mislabeled data which nevertheless were predicted as true label with ResNet18 on CIFAR100. We refer to this as “Oracle bound”. See text for more details. The bound predicted by RATT (RHS in (8.5)) is vacuous.

G.3.4 Hyperparameter Details

Fig. 8.1 We use clean training dataset of size $40,000$. We fix the amount of unlabeled data at 20% of the clean size, i.e. we include additional $8,000$ points with randomly assigned labels. We use test set of $10,000$ points. For both MLP and ResNet, we use SGD with an initial learning rate of 0.1 and momentum 0.9. We fix the weight decay parameter at 5×10^{-4}. After 100 epochs, we decay the learning rate to 0.01. We use SGD batch size of 100.

Fig. 8.2 (a) We obtain a toy dataset according to the process described in Sec. G.3.1. We fix $d = 100$ and create a dataset of $50,000$ points with balanced classes. Moreover, we sample additional covariates with the same procedure to create randomly labeled dataset. For both SGD and GD training, we use a fixed learning rate 0.1.

Fig. 8.2 (b) Similar to binary CIFAR, we use clean training dataset of size $40,000$ and fix the amount of unlabeled data at 20% of the clean dataset size. To train wide nets, we use a fixed learning of 0.001 with GD and SGD. We decide the weight decay parameter and the early stopping point that maximizes our generalization bound (i.e. without peeking at unseen data). We use SGD batch size of 100.

Fig. 8.2 (c) With IMDb dataset, we use a clean dataset of size $20,000$ and as before, fix the amount of unlabeled data at 20% of the clean data. To train ELMo model, we use Adam optimizer with a fixed learning rate 0.01 and weight decay 10^{-6} to minimize cross entropy loss. We train with batch size 32 for 3 epochs. To fine-tune BERT model, we use Adam optimizer with learning rate 5×10^{-5} to minimize cross entropy loss. We train with a batch size of 16 for 1 epoch.

Table 8.1 For multiclass datasets, we train both MLP and ResNet with the same hyperparameters as described before. We sample a clean training dataset of size $40,000$ and fix the amount of unlabeled data at 20% of the clean size. We use SGD with an initial learning rate of 0.1 and momentum 0.9. We fix the weight decay parameter at 5×10^{-4}. After 30 epochs for ResNet and after 50 epochs for MLP, we decay the learning rate to 0.01. We use SGD with batch size 100. For Fig. G.5, we use the same hyperparameters as CIFAR10 training, except we now decay learning rate after 100 epochs.

In all experiments, to identify the best possible accuracy on just the clean data, we use the exact same set of hyperparamters except the stopping point. We choose a stopping point that maximizes test performance.

G.3.5 Summary of experiments

Classification type	Model category	Model	Dataset
Binary	Low dimensional	Linear model	Toy Gaussain dataset
	Overparameterized linear nets	2-layer wide net	Binary MNIST
	Deep nets	MLP	Binary MNIST
			Binary CIFAR
		ResNet	Binary MNIST
			Binary CIFAR
		ELMo-LSTM model	IMDb Sentiment Analysis
		BERT pre-trained model	IMDb Sentiment Analysis
Multiclass	Deep nets	MLP	MNIST
			CIFAR10
		ResNet	MNIST
			CIFAR10
			CIFAR100

G.4 Proof of Lemma G.2.2

Proof of Lemma G.2.2. Recall, we have a training set $S \cup \widetilde{S}_C$. We defined leave-one-out error on mislabeled points as

$$\mathcal{E}_{\text{LOO}(\widetilde{S}_M)} = \frac{\sum_{(x_i,y_i)\in\widetilde{S}_M} \mathcal{E}(f_{(i)}(x_i), y_i)}{\left|\widetilde{S}_M\right|} \,,$$

where $f_{(i)} := f(\mathcal{A}, (S \cup \widetilde{S})_{(i)})$. Define $S' := S \cup \widetilde{S}$. Assume (x, y) and (x', y') as i.i.d. samples from $\mathcal{D}'$. Using Lemma 25 in Bousquet and Elisseeff (2002), we have

$$\begin{aligned}\mathbb{E}\left[\left(\mathcal{E}_{\mathcal{D}'}(\widehat{w}f) - \mathcal{E}_{\text{LOO}(\widetilde{S}_M)}\right)^2\right] \leqslant & \mathbb{E}_{S',(x,y),(x',y')}\left[\mathcal{E}(\widehat{w}f(x), y)\mathcal{E}(\widehat{w}f(x'), y')\right] - 2\mathbb{E}_{S',(x,y)}\left[\mathcal{E}(\widehat{w}f(x), y)\mathcal{E}(f_{(i)}(x_i), y_i\right. \\ & + \frac{m_1 - 1}{m_1}\mathbb{E}_{S'}\left[\mathcal{E}(f_{(i)}(x_i), y_i)\mathcal{E}(f_{(j)}(x_j), y_j)\right] + \frac{1}{m_1}\mathbb{E}_{S'}\left[\mathcal{E}(f_{(i)}(x_i), y_i)\right] \,.\end{aligned} \tag{G.114}$$

We can rewrite the equation above as :

$$\begin{aligned}\mathbb{E}\left[\left(\mathcal{E}_{\mathcal{D}'}(\widehat{w}f) - \mathcal{E}_{\text{LOO}(\widetilde{S}_M)}\right)^2\right] \leqslant & \underbrace{\mathbb{E}_{S',(x,y),(x',y')}\left[\mathcal{E}(\widehat{w}f(x), y)\mathcal{E}(\widehat{w}f(x'), y') - \mathcal{E}(\widehat{w}f(x), y)\mathcal{E}(f_{(i)}(x_i), y_i)\right]}_{\text{I}} \\ & + \underbrace{\mathbb{E}_{S'}\left[\mathcal{E}(f_{(i)}(x_i), y_i)\mathcal{E}(f_{(j)}(x_j), y_j) - \mathcal{E}(\widehat{w}f(x), y)\mathcal{E}(f_{(i)}(x_i), y_i)\right]}_{\text{II}} \\ & + \underbrace{\frac{1}{m_1}\mathbb{E}_{S'}\left[\mathcal{E}(f_{(i)}(x_i), y_i) - \mathcal{E}(f_{(i)}(x_i), y_i)\mathcal{E}(f_{(j)}(x_j), y_j)\right]}_{\text{III}} \,.\end{aligned} \tag{G.115}$$

We will now bound term III. Using Cauchy-Schwarz's inequality, we have

$$\mathbb{E}_{S'}\left[\mathcal{E}(f_{(i)}(x_i), y_i) - \mathcal{E}(f_{(i)}(x_i), y_i)\mathcal{E}(f_{(j)}(x_j), y_j)\right]^2 \leqslant \mathbb{E}_{S'}\left[\mathcal{E}(f_{(i)}(x_i), y_i)\right]^2 \mathbb{E}_{S'}\left[1 - \mathcal{E}(f_{(j)}(x_j), y_j)\right]^2 \tag{G.116}$$

$$\leqslant \frac{1}{4} \,. \tag{G.117}$$

Note that since (x_i, y_i), (x_j, y_j), (x, y), and (x', y') are all from same distribution $\mathcal{D}'$, we directly incorporate the bounds on term I and II from the proof of Lemma 9 in Bousquet and Elisseeff (2002). Combining that with (G.117) and our definition of hypothesis stability in Condition 8.4.1, we have the required claim.

□

Appendix H

Appendix: Leveraging Unlabeled Data to Predict Out-of-Distribution Performance

H.1 Proofs from Sec. 12.2

Before proving results from Sec. 12.2, we introduce some notations. Define $\mathcal{E}(f(x), y) := \mathbb{I}\left[y \notin \arg\max_{j \in \mathcal{Y}} f_j(x)\right]$. We express the *population error* on distribution $\mathcal{D}$ as $\mathcal{E}_{\mathcal{D}}(f) := \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathcal{E}(f(x), y)\right]$.

Proof of Proposition 9.3.1. Consider a binary classification problem. Assume $\mathscr{P}$ be the set of possible target conditional distribution of labels given $p_s(x, y)$ and $p_t(x)$.

The forward direction is simple. If $\mathscr{P} = \{p_t(y|x)\}$ is singleton given $p_s(x, y)$ and $p_t(x)$, then the error of any classifier f on the target domain is identified and is given by

$$\mathcal{E}_{\mathcal{D}^T}(f) = \mathbb{E}_{x\sim p_t(x), y\sim p_t(y|x)}\left[\mathbb{I}\left[\arg\max_{j\in\mathcal{Y}} f_j(x) \neq y\right]\right] . \tag{H.1}$$

For the reverse direction assume that given $p_t(x)$ and $p_s(x, y)$, we have two possible distributions $\mathcal{D}^T$ and $\mathcal{D}^{T'}$ with $p_t(y|x), p'_t(y|x) \in \mathscr{P}$ such that on some x with $p_t(x) > 0$, we have $p_t(y|x) \neq p'_t(y|x)$. Consider $\mathcal{X}_M = \{x \in \mathcal{X} | p_t(x) > 0 \text{ and } p_t(y = 1|x) \neq p'_t(y = 1|x)\}$ be the set of all input covariates where the two distributions differ. We will now choose a classifier f such that the error on the two distributions differ. On a subset $\mathcal{X}^1_M = \{x \in \mathcal{X} | p_t(x) > 0 \text{ and } p_t(y = 1|x) > p'_t(y = 1|x)\}$, assume $f(x) = 0$ and on a subset $\mathcal{X}^2_M = \{x \in \mathcal{X} | p_t(x) > 0 \text{ and } p_t(y = 1|x) < p'_t(y = 1|x)\}$, assume $f(x) = 1$. We will show that the error of f on distribution with $p_t(y|x)$ is strictly greater than the error of f on distribution with $p'_t(y|x)$. Formally,

$\mathcal{E}_{\mathcal{D}^T}(f) - \mathcal{E}_{\mathcal{D}^{T'}}(f)$

$$
\begin{aligned}
&= \mathbb{E}_{x\sim p_t(x), y\sim p_t(y|x)}\left[\mathbb{I}\left[\arg\max_{j\in\mathcal{Y}} f_j(x) \neq y\right]\right] - \mathbb{E}_{x\sim p_t(x), y\sim p_t'(y|x)}\left[\mathbb{I}\left[\arg\max_{j\in\mathcal{Y}} f_j(x) \neq y\right]\right] \\
&= \int_{x\in\mathcal{X}_M} \mathbb{I}\left[f(x)\neq 0\right]\left(p_t(y=0|x) - p_t'(y=0|x)\right)p_t(x)dx \\
&\quad + \int_{x\in\mathcal{X}_M} \mathbb{I}\left[f(x)\neq 1\right]\left(p_t(y=1|x) - p_t'(y=1|x)\right)p_t(x)dx \\
&= \int_{x\in\mathcal{X}_M^2} \left(p_t(y=0|x) - p_t'(y=0|x)\right)p_t(x)dx + \int_{x\in\mathcal{X}_M^1} \left(p_t(y=1|x) - p_t'(y=1|x)\right)p_t(x)dx \\
&> 0\,, \qquad\qquad \text{(H.2)}
\end{aligned}
$$

where the last step follows by construction of the set $\mathcal{X}_M^1$ and $\mathcal{X}_M^2$. Since $\mathcal{E}_{\mathcal{D}^T}(f) \neq \mathcal{E}_{\mathcal{D}^{T'}}(f)$, given the information of $p_t(x)$ and $p_s(x,y)$ it is impossible to distinguish the two values of the error with classifier f. Thus, we obtain a contradiction on the assumption that $p_t(y|x) \neq p_t'(y|x)$. Hence, we must pose restrictions on the nature of shift such that $\mathscr{P}$ is singleton to to identify accuracy on the target. □

Proof of Corollary 9.3.2. The corollary follows directly from Proposition 9.3.1. Since two different target conditional distribution can lead to different error estimates without assumptions on the classifier, no method can estimate two different quantities from the same given information. We illustrate this in Example 1 next. □

H.2 Estimating accuracy in covariate shift or label shift

Accuracy estimation under covariate shift assumption Under the assumption that $p_t(y|x) = p_s(y|x)$, accuracy on the target domain can be estimated as follows:

$$
\begin{aligned}
\mathcal{E}_{\mathcal{D}^{\mathrm{T}}}(f) &= \mathbb{E}_{(x,y)\sim\mathcal{D}^{\mathrm{S}}}\left[\frac{p_t(x,y)}{p_s(x,y)}\mathbb{I}\left[f(x)\neq y\right]\right] && \text{(H.3)} \\
&= \mathbb{E}_{(x,y)\sim\mathcal{D}^{\mathrm{S}}}\left[\frac{p_t(x)}{p_s(x)}\mathbb{I}\left[f(x)\neq y\right]\right]. && \text{(H.4)}
\end{aligned}
$$

Given access to $p_t(x)$ and $p_s(x)$, one can directly estimate the expression in (H.4).

Accuracy estimation under label shift assumption Under the assumption that $p_t(x|y) = p_s(x|y)$, accuracy on the target domain can be estimated as follows:

$$
\begin{aligned}
\mathcal{E}_{\mathcal{D}^{\mathrm{T}}}(f) &= \mathbb{E}_{(x,y)\sim\mathcal{D}^{\mathrm{S}}}\left[\frac{p_t(x,y)}{p_s(x,y)}\mathbb{I}\left[f(x)\neq y\right]\right] && \text{(H.5)} \\
&= \mathbb{E}_{(x,y)\sim\mathcal{D}^{\mathrm{S}}}\left[\frac{p_t(y)}{p_s(y)}\mathbb{I}\left[f(x)\neq y\right]\right]. && \text{(H.6)}
\end{aligned}
$$

Estimating importance ratios $p_t(x)/p_s(x)$ is straightforward under covariate shift assumption when the distributions $p_t(x)$ and $p_s(x)$ are known. For label shift, one can leverage moment matching approach called BBSE (Lipton et al., 2018b) or likelihood minimization approach MLLS (Garg et al., 2020b). Below we discuss the objective of MLLS:

$$w = \arg\max_{w \in \mathcal{W}} \mathbb{E}_{x \sim p_t(x)} \left[\log p_s(y|x)^T w \right] , \tag{H.7}$$

where $\mathcal{W} = \{w \mid \forall y \,, w_y \geqslant 0 \text{ and } \sum_{y=1}^{k} w_y p_s(y) = 1\}$. MLLS objective is guaranteed to obtain consistent estimates for the importance ratios $w^*(y) = p_t(y)/p_s(y)$ under the following condition.

Theorem H.2.1 (Theorem 1 (Garg et al., 2020b))**.** *If the distributions $\{p(x)|y) : y = 1, \ldots, k\}$ are strictly linearly independent, then w^* is the unique maximizer of the MLLS objective* (H.7)*.*

We refer interested reader to Garg et al. (2020b) for details.

Above results of accuracy estimation under label shift and covariate shift can be extended to a generalized label shift and covariate shift settings. Assume a function $h : \mathcal{X} \to \mathcal{Z}$ such that y is independent of x given $h(x)$. In other words $h(x)$ contains all the information needed to predict label y. With help of h, we can extend estimation to following settings: (i) *Generalized covariate shift*, i.e., $p_s(y|h(x)) = p_t(y|h(x))$ and $p_s(h(x)) > 0$ for all $x \in \mathcal{X}_t$; (ii) *Generalized label shift*, i.e., $p_s(h(x)|y) = p_t(h(x)|y)$ and $p_s(y) > 0$ for all $y \in \mathcal{Y}_t$. By simply replacing x with $h(x)$ in (H.4) and (H.7), we will obtain consistent error estimates under these generalized conditions.

Proof of Example 1. Under covariate shift using (H.4), we get

$$\begin{aligned}
\mathcal{E}_1 &= \mathbb{E}_{(x,y) \sim p_s(x,y)} \left[\frac{p_t(x)}{p_s(x)} \mathbb{I}\left[f(x) \neq y \right] \right] \\
&= \mathbb{E}_{x \sim p_s(x, y=0)} \left[\frac{p_t(x)}{p_s(x)} \mathbb{I}\left[f(x) \neq 0 \right] \right] + \mathbb{E}_{x \sim p_s(x, y=1)} \left[\frac{p_t(x)}{p_s(x)} \mathbb{I}\left[f(x) \neq 1 \right] \right] \\
&= \int \mathbb{I}\left[f(x) \neq 0 \right] p_t(x) p_s(y=0|x) dx + \int \mathbb{I}\left[f(x) \neq 1 \right] p_t(x) p_s(y=1|x) dx
\end{aligned}$$

Under label shift using (H.6), we get

$$\begin{aligned}
\mathcal{E}_2 &= \mathbb{E}_{(x,y) \sim \mathcal{D}^{\mathrm{S}}} \left[\frac{p_t(y)}{p_s(y)} \mathbb{I}\left[f(x) \neq y \right] \right] \\
&= \mathbb{E}_{x \sim p_s(x, y=0)} \left[\frac{\beta}{\alpha} \mathbb{I}\left[f(x) \neq 0 \right] \right] + \mathbb{E}_{x \sim p_s(x, y=1)} \left[\frac{1-\beta}{1-\alpha} \mathbb{I}\left[f(x) \neq 1 \right] \right] \\
&= \int \mathbb{I}\left[f(x) \neq 0 \right] \frac{\beta}{\alpha} p_s(y=0|x) p_s(x) dx + \int \mathbb{I}\left[f(x) \neq 1 \right] \frac{(1-\beta)}{(1-\alpha)} p_s(y=1|x) p_s(x) dx
\end{aligned}$$

Then $\mathcal{E}_1 - \mathcal{E}_2$ is given by

$$\mathcal{E}_1 - \mathcal{E}_2 = \int \mathbb{I}\left[f(x) \neq 0 \right] p_s(y=0|x) \left[p_t(x) - \frac{\beta}{\alpha} p_s(x) \right] dx$$

$$
\begin{aligned}
&+ \int \mathbb{I}\left[f(x) \neq 1\right] p_s(y=1|x) \left[p_t(x) - \frac{(1-\beta)}{(1-\alpha)} p_s(x)\right] dx \\
&= \int \mathbb{I}\left[f(x) \neq 0\right] p_s(y=0|x) \frac{(\alpha - \beta)}{\alpha} \phi(\mu_2) dx \\
&+ \int \mathbb{I}\left[f(x) \neq 1\right] p_s(y=1|x) \frac{(\alpha - \beta)}{1-\alpha} \phi(\mu_1) dx \,.
\end{aligned} \tag{H.8}
$$

If $\alpha > \beta$, then $\mathcal{E}_1 > \mathcal{E}_2$ and if $\alpha < \beta$, then $\mathcal{E}_1 < \mathcal{E}_2$. Since $\mathcal{E}_1 \neq \mathcal{E}_2$ for arbitrary f, given access to $p_s(x,y)$, and $p_t(x)$, any method that consistently estimates error under covariate shift will give an incorrect estimate under label shift and vice-versa. The reason being that the same $p_t(x)$ and $p_s(x,y)$ can correspond to error $\mathcal{E}_1$ (under covariate shift) or error $\mathcal{E}_2$ (under label shift) either of which is not discernable absent further assumptions on the nature of shift. □

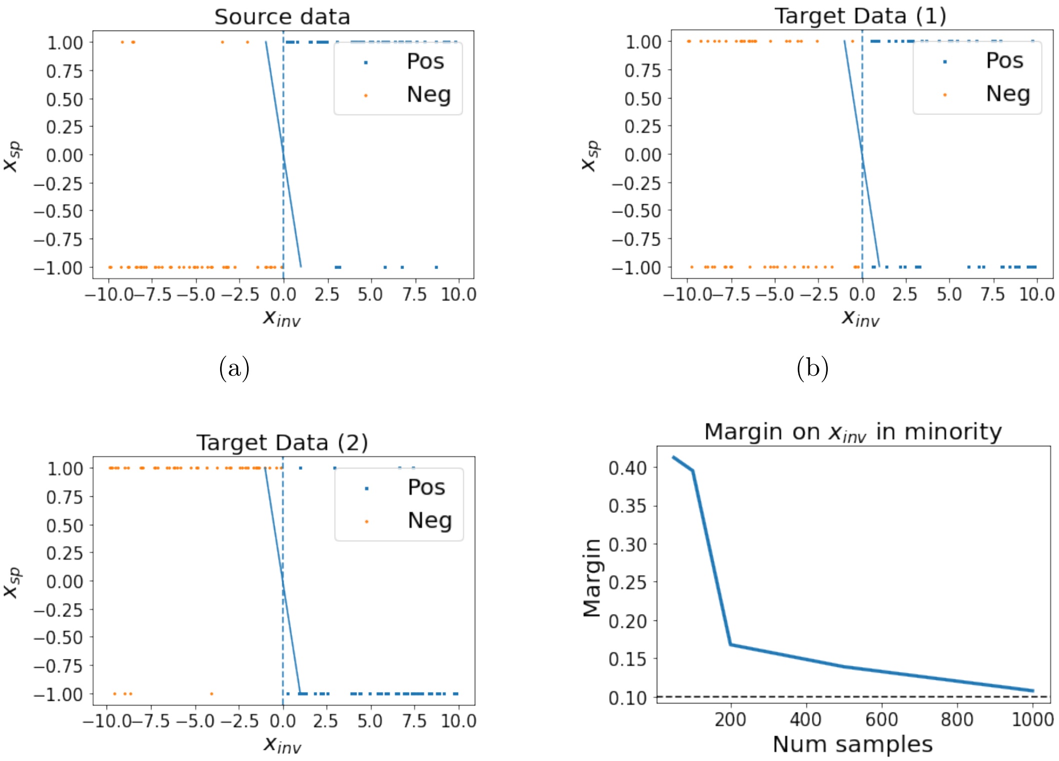
H.3 Alternate interpretation of ATC

Consider the following framework: Given a datum (x, y), define a binary classification problem of whether the model prediction $\arg\max f(x)$ was correct or incorrect. In particular, if the model prediction matches the true label, then we assign a label 1 (positive) and conversely, if the model prediction doesn't match the true label then we assign a label 0 (negative).

Our method can be interpreted as identifying examples for correct and incorrect prediction based on the value of the score function $s(f(x))$, i.e., if the score $s(f(x))$ is greater than or equal to the threshold t then our method predicts that the classifier correctly predicted datum (x, y) and vice-versa if the score is less than t. A method that can solve this task will perfectly estimate the target performance. However, such an expectation is unrealistic. Instead, ATC expects that *most* of the examples with score above threshold are correct and most of the examples below the threshold are incorrect. More importantly, ATC selects a threshold such that the number of falsely identified correct predictions match falsely identified incorrect predictions on source distribution, thereby balancing incorrect predictions. We expect useful estimates of accuracy with ATC if the threshold transfers to target, i.e. if the number of falsely identified correct predictions match falsely identified incorrect predictions on target. This interpretation relates our method to the OOD detection literature where Hendrycks and Gimpel (2017); Hendrycks et al. (2019) highlight that classifiers tend to assign higher confidence to in-distribution examples and leverage maximum softmax confidence (or logit) to perform OOD detection.

H.4 Details on the Toy Model

Skews observed in this toy model In Fig. H.1, we illustrate the toy model used in our empirical experiment. In the same setup, we empirically observe that the margin on population with less density is large, i.e., margin is much greater than γ when the number



(a)

(b)

(c)

(d)

Figure H.1: Illustration of toy model. (a) Source data at $n = 100$. (b) Target data with $p'_s = 0.5$. (b) Target data with $p'_s = 0.9$. (c) Margin of x_{inv} in the minority group in source data. As sample size increases the margin saturates to true margin $\gamma = 0.1$.

of observed samples is small (in Fig. H.1 (d)). Building on this observation, Nagarajan et al. (2020) showed in cases when margin decreases with number of samples, a max margin classifier trained on finite samples is bound to depend on the spurious features in such cases. They referred to this skew as *geometric skew*.

Moreover, even when the number of samples are large so that we do not observe geometric skews, Nagarajan et al. (2020) showed that training for finite number of epochs, a linear classifier will have a non zero dependency on the spurious feature. They referred to this skew as *statistical skew*. Due both of these skews, we observe that a linear classifier obtained with training for finite steps on training data with finite samples, will have a non-zero dependency on the spurious feature. We refer interested reader to Nagarajan et al. (2020) for more details.

Proof of Theorem 9.6.1 Recall, we consider a easy-to-learn binary classification problem with two features $x = [x_{-1}, x_{\text{sp}}] \in \mathbb{R}^2$ where x_{inv} is fully predictive invariant feature with a margin $\gamma > 0$ and $x_{\text{sp}} \in \{-1, 1\}$ is a spurious feature (i.e., a feature that is correlated but not predictive of the true label). Conditional on y, the distribution over x_{inv} is given as follows:

$$x_{\text{inv}}|y \sim \begin{cases} U[\gamma, c] & y = 1 \\ U[-c, -\gamma] & y = -1 \end{cases}, \tag{H.9}$$

where c is a fixed constant greater than γ. For simplicity, we assume that label distribution on source is uniform on $\{-1, 1\}$. x_{sp} is distributed such that $P_s[x_{\text{sp}} \cdot (2y - 1) > 0] = p_{\text{sp}}$, where $p_{\text{sp}} \in (0.5, 1.0)$ controls the degree of spurious correlation. To model distribution shift, we simulate target data with different degree of spurious correlation, i.e., in target distribution $P_t[x_{\text{sp}} \cdot (2y - 1) > 0] = p'_{\text{sp}} \in [0, 1]$. Note that here we do not consider shifts in the label distribution but our result extends to arbitrary shifts in the label distribution as well.

In this setup, we examine linear sigmoid classifiers of the form $f(x) = \left[\frac{1}{1+e^{w^T x}}, \frac{e^{w^T x}}{1+e^{w^T x}}\right]$ where $w = [w_{-1}, w_{\text{sp}}] \in \mathbb{R}^2$. We show that given a linear classifier that relies on the spurious feature and achieves a non-trivial performance on the source (i.e., $w_{\text{inv}} > 0$), ATC with maximum confidence score function *consistently* estimates the accuracy on the target distribution. Define $X_M = \{x | x_{\text{sp}} \cdot (2y - 1) < 0\}$ and $X_C = \{x | x_{\text{sp}} \cdot (2y - 1) > 0\}$. Notice that in target distributions, we are changing the fraction of examples in X_M and X_C but we are not changing the distribution of examples within individual set.

Theorem H.4.1. *Given any classifier f with $w_{inv} > 0$ in the above setting, assume that the threshold t is obtained with finite sample approximation of (9.1), i.e., t is selected such that*[1]

$$\sum_{i=1}^{n} \left[\mathbb{I}\left[\max_{j \in \mathcal{Y}} f_j(x_i) < t\right]\right] = \sum_{i=1}^{n} \left[\mathbb{I}\left[\arg\max_{j \in \mathcal{Y}} f_j(x_i) \neq y_i\right]\right], \tag{H.10}$$

where $\{(x_i, y_i)\}_{i=1}^n \sim (\mathcal{D}^S)^n$ are n samples from source distribution. Fix a $\delta > 0$. Assuming $n \geqslant 2\log(4/\delta)/(1 - p_{sp})^2$, then the estimate of accuracy by ATC as in (9.2) satisfies the following with probability at least $1 - \delta$,

$$\left|\mathbb{E}_{x \sim \mathcal{D}^T}\left[\mathbb{I}\left[s(f(x)) < t\right]\right] - \mathbb{E}_{(x,y) \sim \mathcal{D}^T}\left[\mathbb{I}\left[\arg\max_{j \in \mathcal{Y}} f_j(x) \neq y\right]\right]\right| \leqslant \sqrt{\frac{\log(8/\delta)}{n \cdot c_{sp}}}, \tag{H.11}$$

where $\mathcal{D}^T$ is any target distribution considered in our setting and $c_{sp} = (1 - p_{sp})$ if $w_{sp} > 0$ and $c_{sp} = p_{sp}$ otherwise.

Proof. First we consider the case of $w_{\text{sp}} > 0$. The proof follows in two simple steps. First we notice that the classifier will make an error only on some points in X_M and the threshold t will be selected such that the fraction of points in X_M with maximum confidence less than the threshold t will match the error of the classifier on X_M. Classifier with $w_{\text{sp}} > 0$ and $w_{-1} > 0$ will classify all the points in X_C correctly. Second, since the distribution of points is not changing within X_M and X_C, the same threshold continues to work for arbitrary

[1] Note that this is possible because a linear classifier with sigmoid activation assigns a unique score to each point in source distribution.

shift in the fraction of examples in X_M, i.e., p'_{sp}.

Note that when $w_{\text{sp}} > 0$, the classifier makes no error on points in X_C and makes an error on a subset $X_{\text{Err}} = \{x | x_{\text{sp}} \cdot (2y-1) < 0 \,\&\, (w_{\text{inv}} x_{\text{inv}} + w_{\text{sp}} x_{\text{sp}}) \cdot (2y-1) \leqslant 0\}$ of X_M, i.e., $X_{\text{Err}} \subseteq X_M$. Consider $X_{\text{thres}} = \{x | \arg\max_{y \in \mathcal{Y}} f_y(x) \leqslant t\}$ as the set of points that obtain a score less than or equal to t. Now we will show that ATC chooses a threshold t such that all points in X_C gets a score above t, i.e., $X_{\text{thres}} \subseteq X_M$. First note that the score of points close to the true separator in X_C, i.e., at $x_1 = (\gamma, 1)$ and $x_2 = (-\gamma, -1)$ match. In other words, score at x_1 matches with the score of x_2 by symmetricity, i.e.,

$$\arg\max_{y \in \mathcal{Y}} f_y(x_1) = \arg\max_{y \in \mathcal{Y}} f_y(x_2) = \frac{e^{w_{\text{inv}}\gamma + w_{\text{sp}}}}{\left(1 + e^{w_{\text{inv}}\gamma + w_{\text{sp}}}\right)} . \tag{H.12}$$

Hence, if $t \geqslant \arg\max_{y \in \mathcal{Y}} f_y(x_1)$ then we will have $|X_{\text{Err}}| < |X_{\text{thres}}|$ which is contradiction violating definition of t as in (H.10). Thus $X_{\text{thres}} \subseteq X_M$.

Now we will relate LHS and RHS of (H.10) with their expectations using Hoeffdings and DKW inequality to conclude (H.11). Using Hoeffdings' bound, we have with probability at least $1 - \delta/4$

$$\left| \sum_{i \in X_M} \frac{\left[\mathbb{I}\left[\arg\max_{j \in \mathcal{Y}} f_j(x_i) \neq y_i\right]\right]}{|X_M|} - \mathbb{E}_{(x,y) \sim \mathcal{D}^{\text{T}}} \left[\mathbb{I} \left[\arg\max_{j \in \mathcal{Y}} f_j(x) \neq y \right] \right] \right| \leqslant \sqrt{\frac{\log(8/\delta)}{2\,|X_M|}} . \tag{H.13}$$

With DKW inequality, we have with probability at least $1 - \delta/4$

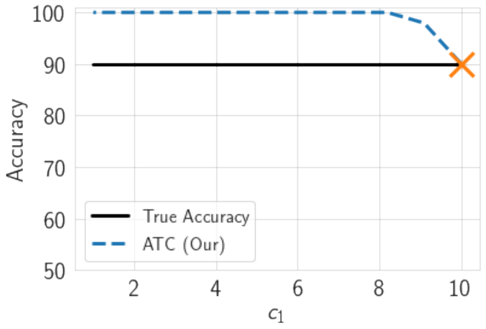
$$\left| \sum_{i \in X_M} \frac{\left[\mathbb{I}\left[\max_{j \in \mathcal{Y}} f_j(x_i) < t'\right]\right]}{|X_M|} - \mathbb{E}_{(x,y) \sim \mathcal{D}^{\text{T}}} \left[\mathbb{I} \left[\max_{j \in \mathcal{Y}} f_j(x) < t' \right] \right] \right| \leqslant \sqrt{\frac{\log(8/\delta)}{2\,|X_M|}} , \tag{H.14}$$

for all $t' > 0$. Combining (H.13) and (H.14) at $t' = t$ with definition (H.10), we have with probability at least $1 - \delta/2$

$$\left| \mathbb{E}_{x \sim \mathcal{D}^{\text{T}}} \left[\mathbb{I}\left[s(f(x)) < t \right] \right] - \mathbb{E}_{(x,y) \sim \mathcal{D}^{\text{T}}} \left[\mathbb{I} \left[\arg\max_{j \in \mathcal{Y}} f_j(x) \neq y \right] \right] \right| \leqslant \sqrt{\frac{\log(8/\delta)}{2\,|X_M|}} . \tag{H.15}$$

Now for the case of $w_{\text{sp}} < 0$, we can use the same arguments on X_C. That is, since now all the error will be on points in X_C and classifier will make no error X_M, we can show that threshold t will be selected such that the fraction of points in X_C with maximum confidence less than the threshold t will match the error of the classifier on X_C. Again, since the distribution of points is not changing within X_M and X_C, the same threshold continues to work for arbitrary shift in the fraction of examples in X_M, i.e., p'_{sp}. Thus with similar arguments, we have

$$\left| \mathbb{E}_{x \sim \mathcal{D}^{\text{T}}} \left[\mathbb{I}\left[s(f(x)) < t \right] \right] - \mathbb{E}_{(x,y) \sim \mathcal{D}^{\text{T}}} \left[\mathbb{I} \left[\arg\max_{j \in \mathcal{Y}} f_j(x) \neq y \right] \right] \right| \leqslant \sqrt{\frac{\log(8/\delta)}{2\,|X_C|}} . \tag{H.16}$$



(a)

Figure H.2: Failure of ATC in our toy model. Shifting the support of target class conditional $p_t(x_{-1}|y)$ may introduce a bias in ATC estimates, e.g., shrinking the support to $c_1(< c)$ (while maintaining uniform distribution) in the target leads to overestimation bias.

Using Hoeffdings' bound, with probability at least $1 - \delta/2$, we have

$$|X_M - n \cdot (1 - p_{\text{sp}})| \leqslant \sqrt{\frac{n \cdot log(4/\delta)}{2}} \,. \tag{H.17}$$

With probability at least $1 - \delta/2$, we have

$$|X_C - n \cdot p_{\text{sp}}| \leqslant \sqrt{\frac{n \cdot log(4/\delta)}{2}} \,. \tag{H.18}$$

Combining (H.17) and (H.15), we get the desired result for $w_{\text{sp}} > 0$. For $w_{\text{sp}} < 0$, we combine (H.18) and (H.16) to get the desired result. □

Issues with IM in toy setting As described in App. **??**, we observe that IM is sensitive to binning strategy. In the main paper, we include IM result with uniform mass binning with 100 bins. Empirically, we observe that we recover the true performance with IM if we use equal width binning with number of bins greater than 5.

Biased estimation with ATC in our toy model We assumed that both in source and target $x_{-1}|y = 1$ is uniform between $[\gamma, c]$ and $x|y = -1$ is uniform between $[-c, -\gamma]$. Shifting the support of target class conditional $p_t(x_{-1}|y)$ may introduce a bias in ATC estimates, e.g., shrinking the support to $c_1(< c)$ (while maintaining uniform distribution) in the target will lead to an over-estimation of the target performance with ATC. We show this failure in Fig. H.2. The reason being that with the same threshold that we see more examples falsely identified as correct as compared to examples falsely identified as incorrect.

H.4.1 A More General Result

Recall, for a given threshold t, we categorize an example (x, y) as a falsely identified correct prediction (ficp) if the predicted label $\widehat{w}y = \arg\max f(x)$ is not the same as y but the predicted score $f_{\widehat{w}y}(x)$ is greater than t. Similarly, an example is falsely identified incorrect prediction (fiip) if the predicted label $\widehat{w}y$ is the same as y but the predicted score $f_{\widehat{w}y}(x)$ is less than t.

In general, we believe that our method will obtain consistent estimates in scenarios where the relative distribution of covariates doesn't change among examples that are falsely identified as incorrect and examples that are falsely identified as correct. In other words, ATC is expected to work if the distribution shift is such that falsely identified incorrect predictions match falsely identified correct prediction.

H.4.2 ATC produces consistent estimate on source distribution

Proposition H.4.2. *Given labeled validation data $\{(x_i, y_i)\}_{i=1}^n$ from a distribution $\mathcal{D}^S$ and a model f, choose a threshold t as in (9.1). Then for $\delta > 0$, with probability at least $1 - \delta$, we have*

$$\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathbb{I}\left[\max_{j\in\mathcal{Y}} f_j(x) < t\right] - \mathbb{I}\left[\arg\max_{j\in\mathcal{Y}} f_j(x) \neq y\right]\right] \leqslant 2\sqrt{\frac{\log(4/\delta)}{2n}} \tag{H.19}$$

Proof. The proof uses (i) Hoeffdings' inequality to relate the accuracy with expected accuracy; and (ii) DKW inequality to show the concentration of the estimated accuracy with our proposed method. Finally, we combine (i) and (ii) using the fact that at selected threshold t the number of false positives is equal to the number of false negatives.

Using Hoeffdings' bound, we have with probability at least $1 - \delta/2$

$$\left|\sum_{i=1}^{n}\left[\mathbb{I}\left[\arg\max_{j\in\mathcal{Y}} f_j(x_i) \neq y_i\right]\right] - \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathbb{I}\left[\arg\max_{j\in\mathcal{Y}} f_j(x) \neq y\right]\right]\right| \leqslant \sqrt{\frac{\log(4/\delta)}{2n}}\,. \tag{H.20}$$

With DKW inequality, we have with probability at least $1 - \delta/2$

$$\left|\sum_{i=1}^{n}\left[\mathbb{I}\left[\max_{j\in\mathcal{Y}} f_j(x_i) < t'\right]\right] - \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathbb{I}\left[\max_{j\in\mathcal{Y}} f_j(x) < t'\right]\right]\right| \leqslant \sqrt{\frac{\log(4/\delta)}{2n}}\,, \tag{H.21}$$

for all $t' > 0$. Finally by definition, we have

$$\sum_{i=1}^{n}\left[\mathbb{I}\left[\max_{j\in\mathcal{Y}} f_j(x_i) < t'\right]\right] = \sum_{i=1}^{n}\left[\mathbb{I}\left[\arg\max_{j\in\mathcal{Y}} f_j(x_i) \neq y_i\right]\right] \tag{H.22}$$

Combining (H.20), (H.21) at $t' = t$, and (H.22), we have the desired result. □

H.5 Basline Methods

Importance-re-weighting (IM) If we can estimate the importance-ratios $\frac{p_t(x)}{p_s(x)}$ with just the unlabeled data from the target and validation labeled data from source, then we can estimate the accuracy as on target as follows:

$$\mathcal{E}_{\mathcal{D}^{\mathrm{T}}}(f) = \mathbb{E}_{(x,y)\sim\mathcal{D}^{\mathrm{S}}}\left[\frac{p_t(x)}{p_s(x)}\mathbb{I}\left[f(x) \neq y\right]\right] . \tag{H.23}$$

As previously discussed, this is particularly useful in the setting of covariate shift (within support) where importance ratios estimation has been explored in the literature in the past. Mandolin (Chen et al., 2021b) extends this approach. They estimate importance-weights with use of extra supervision about the axis along which the distribution is shifting.

In our work, we experiment with uniform mass binning and equal width binning with the number of bins in $[5, 10, 50]$. Overall, we observed that equal width binning works the best with 10 bins. Hence throughout this paper we perform equal width binning with 10 bins to include results with IM.

Average Confidence (AC) If we expect the classifier to be argmax calibrated on the target then average confidence is equal to accuracy of the classifier. Formally, by definition of argmax calibration of f on any distribution $\mathcal{D}$, we have

$$\mathcal{E}_{\mathcal{D}}(f) = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathbb{I}\left[y \notin \arg\max_{j\in\mathcal{Y}} f_j(x)\right]\right] = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\max_{j\in\mathcal{Y}} f_j(x)\right] . \tag{H.24}$$

Difference Of Confidence We estimate the error on target by subtracting difference of confidences on source and target (as a distributional distance (Guillory et al., 2021)) from expected error on source distribution, i.e, $\mathrm{DOC}_{\mathcal{D}^{\mathrm{T}}} = \mathbb{E}_{x\sim\mathcal{D}^{\mathrm{S}}}\left[\mathbb{I}\left[\arg\max_{j\in\mathcal{Y}} f_j(x) \neq y\right]\right] + \mathbb{E}_{x\sim\mathcal{D}^{\mathrm{T}}}\left[\max_{j\in\mathcal{Y}} f_j(x)\right] - \mathbb{E}_{x\sim\mathcal{D}^{\mathrm{S}}}\left[\max_{j\in\mathcal{Y}} f_j(x)\right]$. This is referred to as DOC-Feat in (Guillory et al., 2021).

Generalized Disagreement Equality (GDE) Jiang et al. (2021) proposed average disagreement of two models (trained on the same training set but with different initialization and/or different data ordering) as a approximate measure of accuracy on the underlying data, i.e.,

$$\mathcal{E}_{\mathcal{D}}(f) = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathbb{I}\left[f(x) \neq f'(x)\right]\right] . \tag{H.25}$$

They show that marginal calibration of the model is sufficient to have expected test error equal to the expected of average disagreement of two models where the latter expectation is also taken over the models used to calculate disagreement.

H.6 Details on the Dataset Setup

In our empirical evaluation, we consider both natural and synthetic distribution shifts. We consider shifts on ImageNet (Russakovsky et al., 2015), CIFAR Krizhevsky and Hinton

Train (Source)	Valid (Source)	Evaluation (Target)
MNIST (train)	MNIST (valid)	USPS, SVHN and Q-MNIST
CIFAR10 (train)	CIFAR10 (valid)	CIFAR10v2, 95 CIFAR10-C datasets (Fog and Motion blur, etc.)
CIFAR100 (train)	CIFAR100 (valid)	95 CIFAR100-C datasets (Fog and Motion blur, etc.)
FMoW (2002-12) (train)	FMoW (2002-12) (valid)	FMoW {(2013-15, 2016-17) × (All, Africa, Americas, Oceania, Asia, and Europe)}
RxRx1 (train)	RxRx1(id-val)	RxRx1 (id-test, OOD-val, OOD-test)
Amazon (train)	Amazon (id-val)	Amazon (OOD-val, OOD-test)
CivilComments (train)	CivilComments (id-val)	CiviComments (8 demographic identities male, female, LGBTQ, Christian, Muslim, other religions, Black, and White)
ImageNet (train)	ImageNet (valid)	3 ImageNetv2 datasets, ImageNet-Sketch, 95 ImageNet-C datasets
ImageNet-200 (train)	ImageNet-200 (valid)	3 ImageNet-200v2 datasets, ImageNet-R, ImageNet200-Sketch, 95 ImageNet200-C datasets
Breeds (train)	Breeds (valid)	Same subpopulations as train but unseen images from natural and synthetic shifts in ImageNet, Novel subpopulations on natural and synthetic shifts

Table H.1: Details of the test datasets considered in our evaluation.

(2009), FMoW-Wilds (Christie et al., 2018), RxRx1-Wilds (Taylor et al., 2019), Amazon-Wilds (Ni et al., 2019), CivilComments-Wilds (Borkan et al., 2019), and MNIST (LeCun et al., 1998) datasets.

ImageNet setup. First, we consider synthetic shifts induced to simulate 19 different visual corruptions (e.g., shot noise, motion blur, pixelation etc.) each with 5 different intensities giving us a total of 95 datasets under ImageNet-C (Hendrycks and Dietterich, 2019). Next, we consider natural distribution shifts due to differences in the data collection process. In particular, we consider 3 ImageNetv2 (Recht et al., 2019b) datasets each using a different strategy to collect test sets. We also evaluate performance on images with artistic renditions of object classes, i.e., ImageNet-R (Hendrycks et al., 2021b) and ImageNet-Sketch (Wang et al., 2019b) with hand drawn sketch images. Note that renditions dataset only contains 200 classes from ImageNet. Hence, in the main paper we include results on ImageNet restricted to these 200 classes, which we call as ImageNet-200, and relegate results on ImageNet with 1k classes to appendix.

We also consider Breeds benchmark (Santurkar et al., 2021) in our evaluation to assess robustness to subpopulation shifts, in particular, to understand how accuracy estimation methods behave when novel subpopulations not observed during training are introduced. Breeds leverages class hierarchy in ImageNet to repurpose original classes to be the subpopulations and defines a classification task on superclasses. Subpopulation shift is induced by directly making the subpopulations present in the training and test distributions disjoint. Overall, Breeds benchmark contains 4 datasets Entity-13, Entity-30, Living-17, Non-living-26, each focusing on different subtrees in the hierarchy. To generate Breeds dataset on top of ImageNet, we use the open source library: https://github.com/MadryLab/BREEDS-Benchmarks. We focus on natural and synthetic shifts as in ImageNet on same and different subpopulations in BREEDs. Thus for both the subpopulation (same or novel), we obtain a total of 99 target datasets.

CIFAR setup. Similar to the ImageNet setup, we consider (i) synthetic shifts (CIFAR-10-C) due to common corruptions; and (ii) natural distribution shift (i.e., CIFARv2 (Recht et al., 2018; Torralba et al., 2008)) due to differences in data collection strategy on on CIFAR-10 (Krizhevsky and Hinton, 2009). On CIFAR-100, we just have synthetic shifts due to common corruptions.

FMoW-Wilds setup. In order to consider distribution shifts faced in the wild, we consider FMoW-wilds (Christie et al., 2018; Koh et al., 2021) from Wilds benchmark, which contains satellite images taken in different geographical regions and at different times. We obtain 12 different OOD target sets by considering images between years 2013–2016 and 2016–2018 and by considering five geographical regions as subpopulations (Africa, Americas, Oceania, Asia, and Europe) separately and together.

RxRx1–Wilds setup. Similar to FMoW, we consider RxRx1-Wilds (Taylor et al., 2019) from Wilds benchmark, which contains image of cells obtained by fluorescent microscopy and the task is to genetic treatments the cells received. We obtain 3 target datasets with shift induced by batch effects which make it difficult to draw conclusions from data across experimental batches.

Amazon-Wilds setup. For natural language task, we consider Amazon-Wilds (Ni et al., 2019) dataset from Wilds benchmark, which contains review text and the task is get a corresponding star rating from 1 to 5. We obtain 2 target datasets by considered shifts induced due to different set of reviewers than the training set.

CivilComments-Wilds setup. We also consider CivilComments-Wilds (Borkan et al., 2019) from Wilds benchmark, which contains text comments and the task is to classify them for toxicity. We obtain 18 target datasets depending on whether a comment mentions each of the 8 demographic identities male, female, LGBTQ, Christian, Muslim, other religions, Black, and White.

MNIST setup. For completeness, we also consider distribution shifts on MNIST (LeCun et al., 1998) digit classification as in the prior work (Deng and Zheng, 2021). We use three real shifted datasets, i.e., USPS (Hull, 1994), SVHN (Netzer et al., 2011a) and QMNIST (Yadav and Bottou, 2019).

H.7 Details on the Experimental Setup

All experiments were run on NVIDIA Tesla V100 GPUs. We used PyTorch (Paszke et al., 2019) for experiments.

Deep nets We consider a 4-layered MLP. The PyTorch code for 4-layer MLP is as follows:

```
 nn.Sequential(nn.Flatten(),
nn.Linear(input_dim, 5000, bias=True),
nn.ReLU(),
nn.Linear(5000, 5000, bias=True),
nn.ReLU(),
nn.Linear(5000, 50, bias=True),
nn.ReLU(),
nn.Linear(50, num_label, bias=True)
)
```

We mainly experiment convolutional nets. In particular, we use ResNet18 (He et al., 2016), ResNet50, and DenseNet121 (Huang et al., 2017) architectures with their default implementation in PyTorch. Whenever we initial our models with pre-trained models, we again use default models in PyTorch.

Hyperparameters and Training details As mentioned in the main text we do not alter the standard training procedures and hyperparameters for each task. We present results at final model, however, we observed that the same results extend to an early stopped model as well. For completeness, we include these details below:

CIFAR10 and CIFAR100 We train DenseNet121 and ResNet18 architectures from scratch. We use SGD training with momentum of 0.9 for 300 epochs. We start with learning rate 0.1 and decay it by multiplying it with 0.1 every 100 epochs. We use a weight decay of $5 \times 10^{-}4$. We use batch size of 200. For CIFAR10, we also experiment with the same models pre-trained on ImageNet.

ImageNet For training, we use Adam with a batch size of 64 and learning rate 0.0001. Due to huge size of ImageNet, we could only train two models needed for GDE for 10 epochs. Hence, for relatively small scale experiments, we also perform experiments on ImageNet subset with 200 classes, which we call as ImageNet-200 with the same training procedure. These 200 classes are the same classes as in ImageNet-R dataset. This not only allows us to train ImageNet for 50 epochs but also allows us to use ImageNet-R in our testbed. On the both the datasets, we observe a similar superioriy with ATC. Note that all the models trained here were initialized with a pre-trained ImageNet model with the last layer replaced with random weights.

FMoW-WILDS For all experiments, we follow Koh et al. (2021) and use two architectures DenseNet121 and ResNet50, both pre-trained on ImageNet. We use the Adam optimizer (Kingma and Ba, 2014) with an initial learning rate of 10^{-4} that decays by 0.96 per epoch, and train for 50 epochs and with a batch size of 64.

RxRx1-WILDS For all experiments, we follow Koh et al. (2021) and use two architectures DenseNet121 and ResNet50, both pre-trained on ImageNet. We use Adam optimizer with a learning rate of $1e-4$ and L2-regularization strength of $1e-5$ with a batch size of 75 for 90 epochs. We linearly increase the learning rate for 10 epochs, then decreasing it following a cosine learning rate schedule. Finally, we pick the model that obtains highest in-distribution validation accuracy.

Amazon-WILDS For all experiments, we follow Koh et al. (2021) and finetuned DistilBERT-base-uncased models (Sanh et al., 2019a), using the implementation from Wolf et al. (2020), and with the following hyperparameter settings: batch size 8; learning rate $1e-5$ with the AdamW optimizer (Loshchilov and Hutter, 2017); L2-regularization strength 0.01; 3 epochs with early stopping; and a maximum number of tokens of 512.

CivilComments-WILDS For all experiments, we follow Koh et al. (2021) and fine-tuned DistilBERT-base-uncased models (Sanh et al., 2019a), using the implementation from Wolf et al. (2020) and with the following hyperparameter settings: batch size 16; learning rate $1e-5$ with the AdamW optimizer (Loshchilov and Hutter, 2017) for 5 epochs; L2-regularization strength 0.01; and a maximum number of tokens of 300.

Living17 and Nonliving26 from BREEDS For training, we use SGD with a batch size of 128, weight decay of 10^{-4}, and learning rate 0.1. Models were trained until convergence. Models were trained for a total of 450 epochs, with 10-fold learning rate drops every 150 epochs. Note that since we want to evaluate models for novel subpopulations no pre-training was used. We train two architectures DenseNet121 and ResNet50.

Entity13 and Entity30 from BREEDS For training, we use SGD with a batch size of 128, weight decay of 10^{-4}, and learning rate 0.1. Models were trained until convergence. Models were trained for a total of 300 epochs, with 10-fold learning rate drops every 100 epochs. Note that since we want to evaluate models for novel subpopulations no pre-training was used. We train two architectures DenseNet121 and ResNet50.

MNIST For MNIST, we train a MLP described above with SGD with momentum 0.9 and learning rate 0.01 for 50 epochs. We use weight decay of 10^{-5} and batch size as 200.

We have a single number for CivilComments because it is a binary classification task. For multiclass problems, ATC-NE and ATC-MC can lead to different ordering of examples when ranked with the corresponding scoring function. Temperature scaling on top can further alter the ordering of examples. The changed ordering of examples yields different thresholds and different accuracy estimates. However for binary classification, the two scoring functions are the same as entropy (i.e. $p\log(p) + (1-p)\log(p)$) has a one-to-one mapping to the max conf for $p \in [0, 1]$. Moreover, temperature scaling also doesn't change the order of points for binary classification problems. Hence for the binary classification problems, both the scoring functions with and without temperature scaling yield the same estimates. We have made this clear in the updated draft.

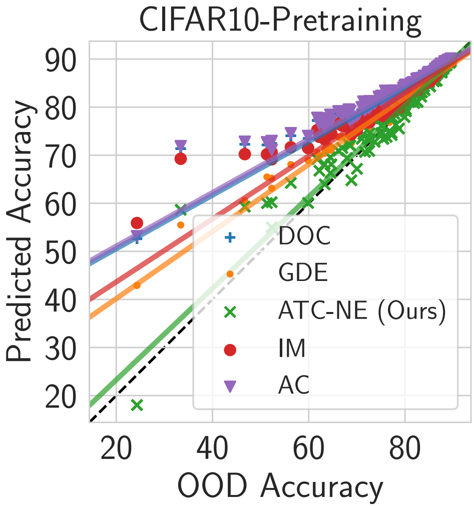
Implementation for Temperature Scaling We use temperature scaling implementation from `https://github.com/kundajelab/abstention`. We use validation set (the same we use to obtain ATC threshold or DOC source error estimate) to tune a single temperature parameter.

H.7.1 Details on Fig. 11.1 (right) setup

For vision datasets, we train a DenseNet model with the exception of FCN model for MNIST dataset. For language datasets, we fine-tune a DistilBERT-base-uncased model. For each of these models, we use the exact same setup as described Sec. H.7. Importantly, to obtain errors on the same scale, we rescale all the errors by subtracting the error of Average Confidence method for each model. Results are reported as mean of the re-scaled errors over 4 seeds.

H.8 Additional Results

H.8.1 CIFAR pretraining Ablation



(a)

Figure H.3: Results with a pretrained DenseNet121 model on CIFAR10. We observe similar behaviour as that with a model trained from scratch.

H.8.2 Breeds results with regression model

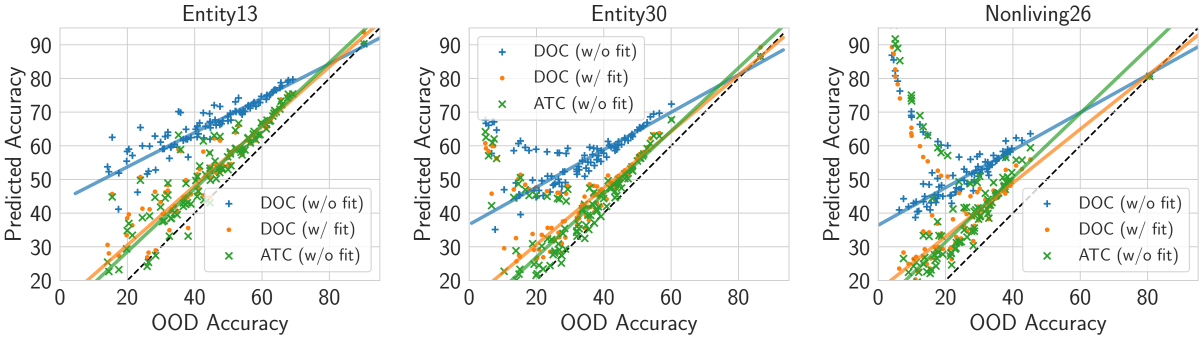


Figure H.4: Scatter plots for DOC with linear fit. Results parallel to Fig. 9.3(Middle) on other Breeds dataset.

Dataset	DOC (w/o fit)	DOC (w fit)	ATC-MC (Ours) (w/o fit)
Living-17	24.32	13.65	**10.07**
Nonliving-26	29.91	**18.13**	19.37
Entity-13	22.18	8.63	8.01
Entity-30	24.71	12.28	**10.21**

Table H.2: *Mean Absolute estimation Error (MAE) results for BREEDs datasets with novel populations in our setup.* After fitting a robust linear model for DOC on same subpopulation, we show predicted accuracy on different subpopulations with fine-tuned DOC (i.e., DOC (w/ fit)) and compare with ATC without any regression model, i.e., ATC (w/o fit). While observe substantial improvements in MAE from DOC (w/o fit) to DOC (w/ fit), ATC (w/o fit) continues to outperform even DOC (w/ fit).

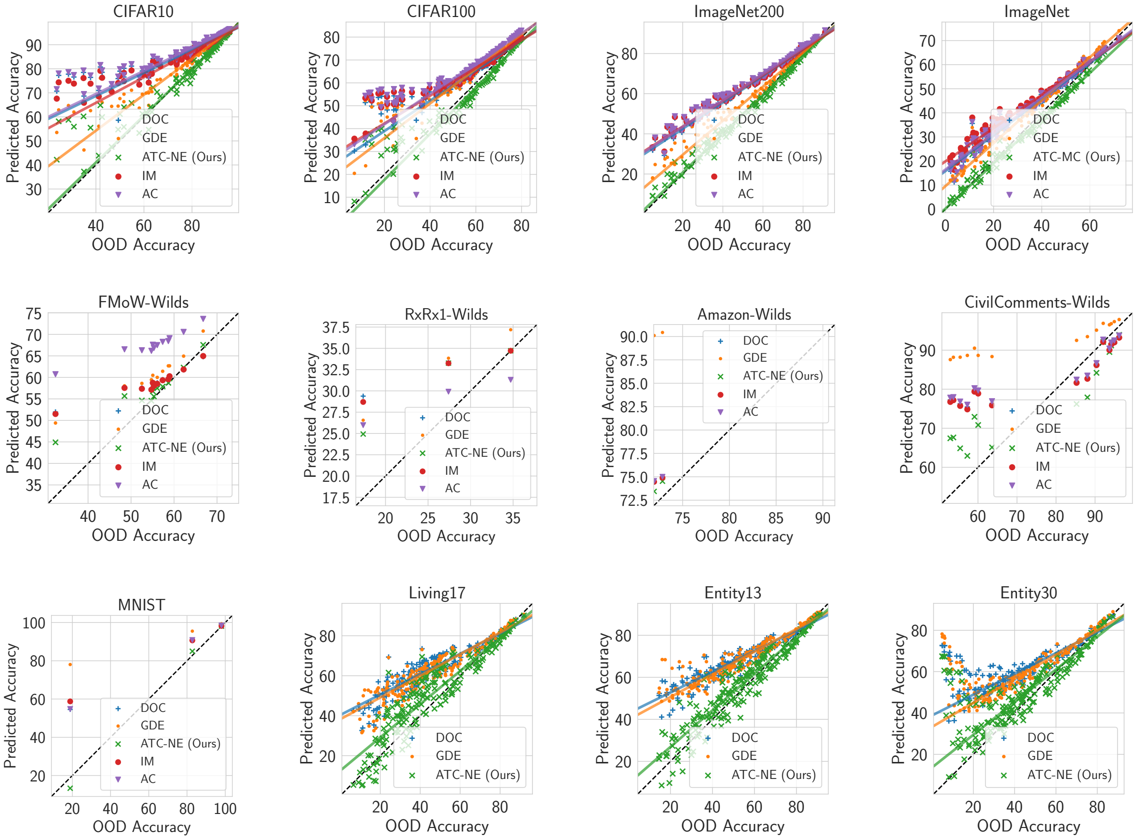


Figure H.5: Scatter plot of predicted accuracy versus (true) OOD accuracy. For vision datasets except MNIST we use a DenseNet121 model. For MNIST, we use a FCN. For language datasets, we use DistillBert-base-uncased. Results reported by aggregating accuracy numbers over 4 different seeds.

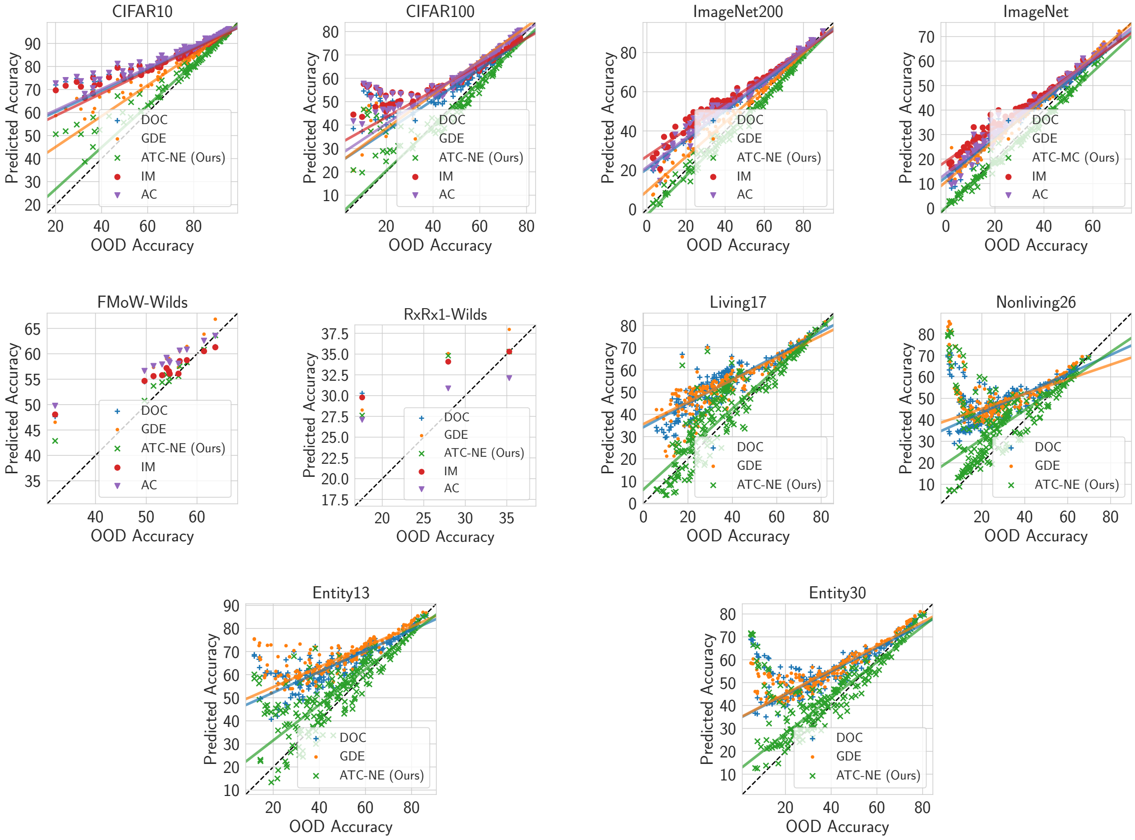


Figure H.6: Scatter plot of predicted accuracy versus (true) OOD accuracy for vision datasets except MNIST with a ResNet50 model. Results reported by aggregating MAE numbers over 4 different seeds.

Appendix I

Appendix: (Almost) Provable Error Bounds Under Distribution Shift via Disagreement Discrepancy

I.1 Comparing Disagreement Losses

We define the alternate losses for maximizing disagreement:

1. Chuang et al. (2020) minimize the negative cross-entropy loss, which is concave in the model logits. That is, they add the term $\log \operatorname{softmax}(h(x)_y)$ to the objective they are minimizing. This loss results in substantially lower disagreement discrepancy than the other two.

2. Pagliardini et al. (2023) use a loss which is not too different from ours. They define the disagreement objective for a point (x, y) as

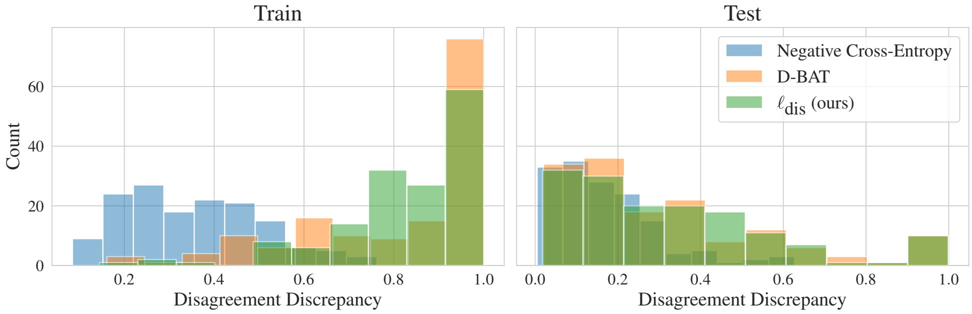
$$\log \left(1 + \frac{\exp(h(x)_y)}{\sum_{\hat{y} \neq y} \exp(h(x)_{\hat{y}})} \right). \tag{I.1}$$

For comparison, ℓ_{dis} can be rewritten as

$$\log \left(1 + \frac{\exp(h(x)_y)}{\exp\left(\frac{1}{|\mathcal{Y}|-1} \sum_{\hat{y} \neq y} h(x)_{\hat{y}} \right)} \right), \tag{I.2}$$

where the incorrect logits are averaged and the exponential is pushed outside the sum. This modification results in (I.2) being convex in the logits and an upper bound to the disagreement 0-1 loss, whereas (I.1) is neither.

Fig. I.1 displays histograms of the achieved disagreement discrepancy across all distributions for each of the disagreement losses (all hyperparameters and random seeds are the same for



Loss	Mean Discrepancy (Train)	Mean Discrepancy (Test)
Neg. X-Ent (Chuang et al., 2020)	$0.3555 \pm .0124$	$0.1694 \pm .0105$
D-BAT (Pagliardini et al., 2023)	$0.8145 \pm .0177$	$0.3224 \pm .0212$
ℓ_{dis} (Ours)	$\underline{0.8333 \pm .0132}$	$\mathbf{0.3322 \pm .0205}$

Figure I.1: Histogram of disagreement discrepancies for each of the three losses, and the average values across all datasets. **Bold** (resp. Underline) indicates the method has higher average discrepancy under a paired t-test at significance $p = .01$ (resp. $p = .025$).

all three losses). The table below it reports the mean disagreement discrepancy on the train and test sets. We find that the negative cross-entropy, being a concave function, results in very low discrepancy. The D-BAT loss (Eq. (I.1)) is reasonably competitive with our loss (Eq. (I.2)) on average, seemingly because it gets very high discrepancy on a subset of shifts. This suggests that it may be particularly suited for a specific type of distribution shift, though it is less good overall. Though the averages are reasonably close, the samples are not independent, so we run a paired t-test and we find that the increases to average train and test discrepancies achieved by ℓ_{dis} are significant at levels $p = 0.024$ and $p = 0.009$, respectively. With enough holdout data, a reasonable approach would be to split the data in two: one subset to validate critics trained on either of the two losses, and another to evaluate the discrepancy of whichever one is ultimately selected.

I.2 Exploration of the Validity Score

To experiment with reducing the complexity of the class $\mathcal{H}$, we evaluate $\textsc{Dis}^2$ on progressively fewer top principal components (PCs) of the features. Precisely, for features of dimension d, we evaluate $\textsc{Dis}^2$ on the same features projected onto their top ${}^{d}/_{k}$ components, for $k \in [1, 4, 16, 32, 64, 128]$ (Fig. I.2). We see that while projecting to fewer and fewer PCs does reduce the error bound value, unlike the logits it is a rather crude way to reduce complexity of $\mathcal{H}$, meaning at some point it goes too far and results in invalid error bounds.

However, during the optimization process we observe that around when this violation occurs, the task of training a critic to both agree on $\mathcal{S}$ and disagree on $\mathcal{T}$ goes from "easy" to "hard". Fig. I.3 shows that on the full features, the critic rapidly ascends to maximum agreement on $\mathcal{S}$, followed by slow decay (due to both overfitting and learning to simultaneously disagree on $\mathcal{T}$). As we drop more and more components, this optimization becomes slower.

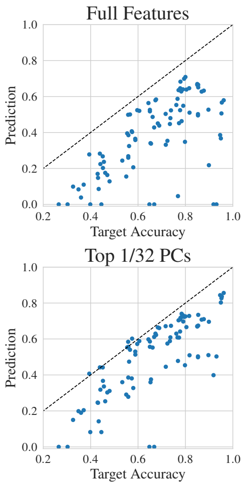


Figure I.2: **Dis^2 bound as fewer principal components are kept.** Reducing the number of top principal components crudely reduces complexity of $\mathcal{H}$—this leads to lower error estimates, but at some point the bounds become invalid for a large fraction of shifts.

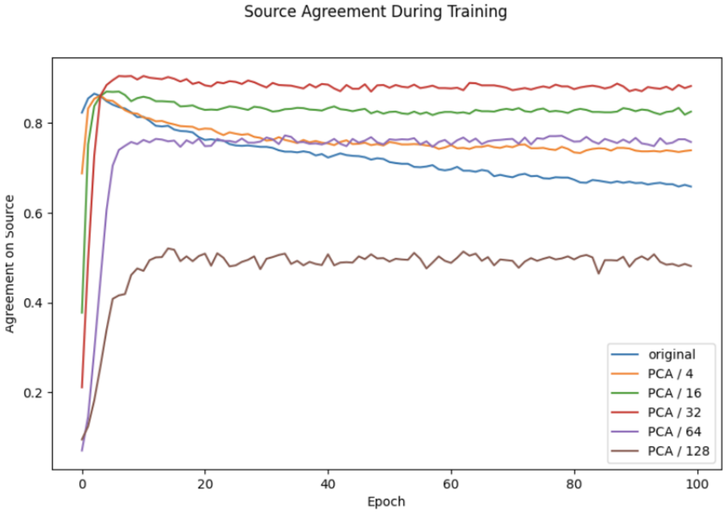


Figure I.3: **Agreement on one shift between $\widehat{h}$ and h' on $\widehat{\mathcal{S}}$ during optimization.** We observe that as the number of top PCs retained drops, the optimization occurs more slowly and less monotonically. For this particular shift, the bound becomes invalid when keeping only the top $^1/_{128}$ components, depicted by the brown line.

We therefore design a "validity score" intended to capture this phenomenon which we refer to as the *cumulative ℓ_1 ratio*. This is defined as the maximum agreement achieved, divided by the cumulative sum of absolute differences in agreement across all epochs up until the maximum was achieved. Formally, let $\{a_i\}_{i=1}^T$ represent the agreement between h' and $\widehat{h}$ after epoch i, i.e. $1 - \epsilon_{\widehat{\mathcal{S}}}(\widehat{h}, h'_i)$, and define $m := \arg\max_{i \in [T]} a_i$. The cumulative ℓ_1 ratio is then $\frac{a_m}{a_1 + \sum_{i=2}^m |a_i - a_{i-1}|}$. Thus, if the agreement rapidly ascends to its maximum without ever going down over the course of an epoch, this ratio will be equal to 1, and if it non-monotonically ascends then the ratio will be significantly less. This definition was simply the first metric we considered which approximately captures the behavior we observed; we expect it could be greatly improved.

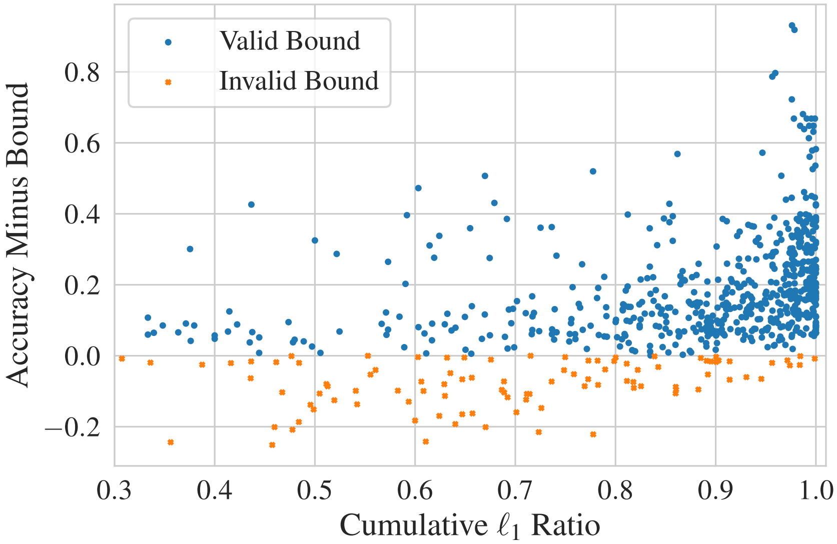


Figure I.4: **Cumulative ℓ_1 ratio versus error prediction gap.** Despite its simplicity, the ratio captures the information encoded in the optimization trajectory, roughly linearly correlating with the tightness and validity of a given prediction. It is thus a useful metric for identifying the ideal number of top PCs to use.

Fig. I.4 displays a scatter plot of the cumulative ℓ_1 ratio versus the difference in estimated and true error for DIS^2 evaluated on the full range of top PCs. A negative value implies that we have underestimated the error (i.e., the bound is not valid). We see that even this very simply metric roughly linearly correlates with the tightness of the bound, which suggests that evaluating over a range of top PC counts and only keeping predictions whose ℓ_1 ratio is above a certain threshold can improve raw predictive accuracy without reducing coverage by too much. Fig. I.5 shows that this is indeed the case: compared to DIS^2 evaluated on the logits, keeping all predictions above a score threshold can produce more accurate error estimates, without *too* severely underestimating error in the worst case.

I.3 Making Baselines More Conservative with LOOCV

To more thoroughly compare DIS^2 to prior estimation techniques, we consider a strengthening of the baselines which may give them higher coverage without too much cost to prediction accuracy. Specifically, for each desired coverage level $\alpha \in [0.9, 0.95, 0.99]$, we use all but one of the datasets to learn a parameter to either scale or shift a method's predictions enough to achieve coverage α. We then evaluate this scaled or shifted prediction on the distribution shifts of the remaining dataset, and we repeat this for each one.

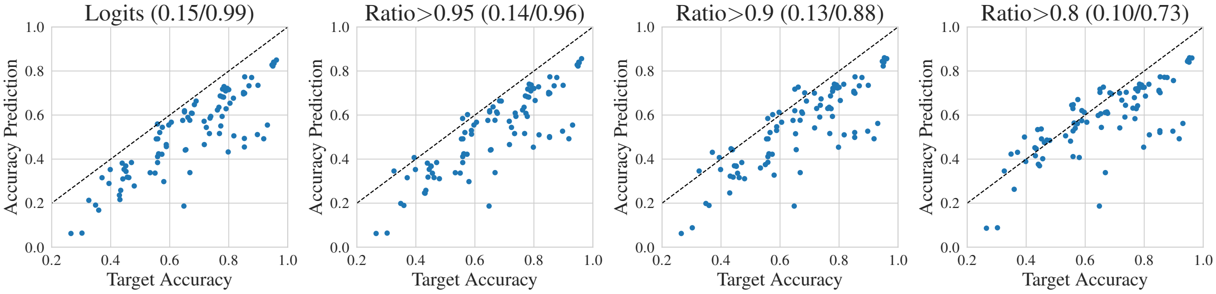


Figure I.5: **Dis^2 bounds and MAE / coverage as the cumulative ℓ_1 ratio threshold is lowered.** Values in parenthesis are (MAE / coverage). By only keeping predictions with ratio above a varying threshold, we can smoothly interpolate between bound validity and raw error prediction accuracy.

The results, found in Table I.1, demonstrate that prior methods can indeed be made to have much higher coverage, although as expected their MAE suffers. Furthermore, they still underestimate error on the tail distribution shifts by quite a bit, and they rarely achieve the desired coverage on the heldout dataset—though they usually come reasonably close. In particular, ATC (Garg et al., 2022c) and COT (Lu et al., 2023) do well with a shift parameter, e.g. at the desired coverage $\alpha = 0.95$ ATC matches Dis^2 in MAE and gets 94.4% coverage (compared to 98.9% by Dis^2). However, its conditional average overestimation is quite high, almost 9%. COT gets much lower overestimation (particularly for higher coverage levels), and it also appears to suffer less on the tail distribution shifts in the sense that $\alpha = 0.99$ does not induce nearly as high MAE as it does for ATC. However, at that level it only achieves 95.6% coverage, and it averages almost 5% accuracy overestimation on the shifts it does not correctly bound (compared to 0.1% by Dis^2). Also, its MAE is still substantially higher than Dis^2, despite getting lower coverage. Finally, we evaluate the scale/shift approach on our Dis^2 bound without the lower order term, but based on the metrics we report there appears to be little reason to prefer it over the untransformed version, one of the baselines, or the original Dis^2 bound.

Taken together, these results imply that if one's goal is predictive accuracy and tail behavior is not important (worst ~10%), ATC or COT will likely get reasonable coverage with a shift parameter—though they still significantly underestimate error on a non-negligible fraction of shifts. If one cares about the long tail of distribution shifts, or prioritizes being conservative at a slight cost to average accuracy, Dis^2 is clearly preferable. Finally, we observe that the randomness which determines which shifts are not correctly bounded by Dis^2 is "decoupled" from the distributions themselves under Theorem 10.2.4, in the sense that it is an artifact of the random samples, rather than a property of the distribution (recall **??**). This is in contrast with the shift/scale approach which would produce almost identical results under larger sample sizes because it does not account for finite sample effects. This implies that some distribution shifts are simply "unsuitable" for prior methods because they do not satisfy whatever condition these methods rely on, and observing more samples will not remedy this problem. It is clear that working to understand these conditions is crucial for reliability and interpretability, since we are not currently able to identify which distributions are suitable a priori.

		MAE (↓)			Coverage (↑)			Overest. (↓)		
	$\alpha \rightarrow$	0.9	0.95	0.99	0.9	0.95	0.99	0.9	0.95	0.99
Method	Adjustment									
AC	none	0.106			0.122			0.118		
	shift	0.153	0.201	0.465	0.878	0.922	0.956	0.119	0.138	0.149
	scale	0.195	0.221	0.416	0.911	0.922	0.967	0.135	0.097	0.145
DoC	none	0.105			0.167			0.122		
	shift	0.158	0.200	0.467	0.878	0.911	0.956	0.116	0.125	0.154
	scale	0.195	0.223	0.417	0.900	0.944	0.967	0.123	0.139	0.139
ATC NE	none	0.067			0.289			0.083		
	shift	0.117	0.150	0.309	0.900	0.944	0.978	0.072	0.088	0.127
	scale	0.128	0.153	0.357	0.889	0.933	0.978	0.062	0.074	0.144
COT	none	0.069			0.256			0.085		
	shift	0.115	0.140	0.232	0.878	0.944	0.956	0.049	0.065	0.048
	scale	0.150	0.193	0.248	0.889	0.944	0.956	0.074	0.066	0.044
Dis^2 (w/o δ)	none	0.083			0.756			0.072		
	shift	0.159	0.169	0.197	0.889	0.933	0.989	0.021	0.010	0.017
	scale	0.149	0.168	0.197	0.889	0.933	0.989	0.023	0.021	0.004
Dis^2 ($\delta = 10^{-2}$)	none	0.150			0.989			0.001		
Dis^2 ($\delta = 10^{-3}$)	none	0.174			1.000			0.000		

Table I.1: MAE, coverage, and conditional average overestimation for the strengthened baselines with a shift or scale parameter on non-domain-adversarial representations. Because a desired coverage α is only used when an adjustment is learned, "none"—representing no adjustment—does not vary with α.

I.4 Proving that Assumption 4 Holds for Some Datasets

Here we describe how the equivalence of Assumption 4 and the bound in Theorem 10.2.4 allow us to prove that the assumption holds with high probability. By repeating essentially the same proof as Theorem 10.2.4 in the other direction, we get the following corollary:

Corollary I.4.1. *If Assumption 4 does* not *hold, then with probability* $\geqslant 1 - \delta$,

$$\epsilon_{\widehat{\mathcal{T}}}(\widehat{h}) > \epsilon_{\widehat{\mathcal{S}}}(\widehat{h}) + \widehat{\Delta}(\widehat{h}, h') - \sqrt{\frac{2(n_S + n_T)\log {}^{1}\!/\!{}_{\delta}}{n_S n_T}}.$$

Note that the concentration term here is different from Theorem 10.2.4 because we are bounding the empirical target error, rather than the true target error. The reason for this change is that now we can make direct use of its contrapositive:

Corollary I.4.2. *With probability $\geqslant 1-\delta$ over the randomness of the samples $\widehat{\mathcal{S}}$ and $\widehat{\mathcal{T}}$, if it is the case that*

$$\epsilon_{\widehat{\mathcal{T}}}(\widehat{h}) \leqslant \epsilon_{\widehat{\mathcal{S}}}(\widehat{h}) + \widehat{\Delta}(\widehat{h}, h') - \sqrt{\frac{2(n_S + n_T)\log {}^1\!/\!_\delta}{n_S n_T}},$$

then Assumption 4 must hold.

We evaluate this bound on non-domain-adversarial shifts with $\delta = 10^{-6}$. As some of the BREEDS shifts have as few as 68 test samples, we restrict ourselves to shifts with $n_T \geqslant 500$ to ignore those where the finite-sample term heavily dominates; this removes a little over 20% of all shifts. Among the remainder, we find that the bound in Theorem I.4.2 holds 55.7% of the time when using full features and 25.7% of the time when using logits. This means that for these shifts, we can be essentially certain that Assumption 4—and therefore also Assumption 3—is true.

Note that the fact that the bound is *not* violated for a given shift does not at all imply that the assumption is not true. In general, the only rigorous way to prove that Assumption 4 does not hold would be to show that for a fixed δ, the fraction of shifts for which the bound in Theorem 10.2.4 does not hold is larger than δ (in a manner that is statistically significant under the appropriate hypothesis test). Because this never occurs in our experiments, we cannot conclude that the assumption is ever false. At the same time, the fact that the bound *does* hold at least $1-\delta$ of the time does not prove that the assumption is true—it merely suggests that it is reasonable and that the bound should continue to hold in the future. This is why it is important for Assumption 4 to be simple and intuitive, so that we can trust that it will persist and anticipate when it will not.

However, Theorem I.4.2 allows us to make a substantially stronger statement. In fact, it says that for *any* distribution shift, with enough samples, we can prove a posteriori whether or not Assumption 4 holds, because the gap between these two bounds will shrink with increasing sample size.

I.5 Fig. 10.1 Stratified by Training Method

I.6 Additional Figures and Discussion

I.6.1 How does DIS^2 Improve over $\mathcal{H}\Delta\mathcal{H}$-Divergence?

Consider the task of learning a linear classifier to discriminate between squares and circles on the source distribution $\mathcal{S}$ (blue) and then bounding the error of this classifier on the target distribution $\mathcal{T}$ (red), whose true labels are unknown and are therefore depicted as triangles. Fig. I.7(a) demonstrates that both $\mathcal{H}$- and $\mathcal{H}\Delta\mathcal{H}$-divergence achieve their maximal value of 1, because both h_1 and $h_2 \oplus h_3$ perfectly discriminate between $\mathcal{S}$ and $\mathcal{T}$. Thus both bounds would be vacuous.

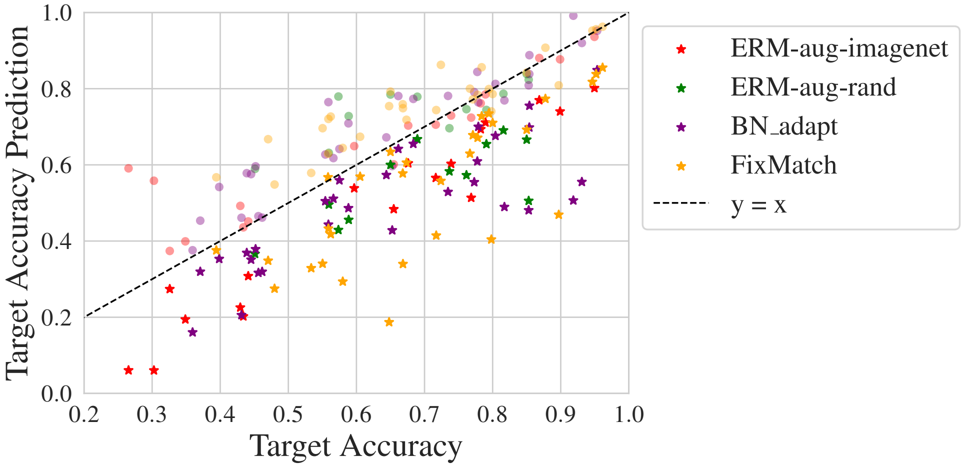


Figure I.6: **Error prediction stratified by training method.** Stars denote Dɪs^2, circles are ATC NE. We see that Dɪs^2 maintains its validity across different training methods.

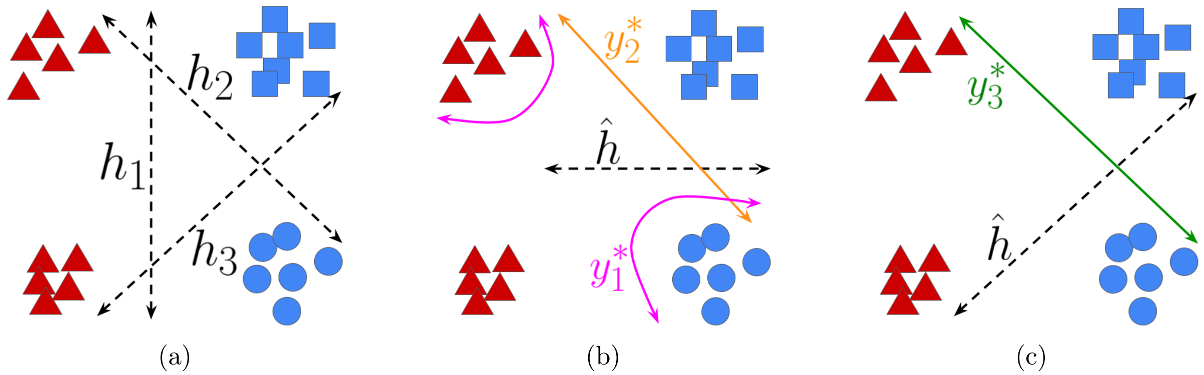


Figure I.7: **The advantage of Dɪs^2 over bounds based on $\mathcal{H}$- and $\mathcal{H}\Delta\mathcal{H}$-divergence.** Consider the task of classifying circles and squares (triangles are unlabeled). **(a):** Because h_1 and $h_2 \oplus h_3$ perfectly discriminate between $\mathcal{S}$ (blue) and $\mathcal{T}$ (red), $\mathcal{H}$- and $\mathcal{H}\Delta\mathcal{H}$-divergence bounds are always vacuous. In contrast, Dɪs^2 is only vacuous when 0% accuracy is induced by a reasonably likely ground truth (such as y_3^* in **(c)**, but not y_1^* in **(b)**), and can often give non-vacuous bounds (such as y_2^* in **(b)**).

Now, suppose we were to learn the max-margin $\hat{h}$ on the source distribution (Fig. I.7(b)). It is *possible* that the true labels are given by the worst-case boundary as depicted by y_1^* (pink), thus "flipping" the labels and causing $\hat{h}$ to have 0 accuracy on $\mathcal{T}$. In this setting, a vacuous bound is correct. However, this seems rather unlikely to occur in practice—instead, recent experimental evidence (Kang et al., 2020; Kirichenko et al., 2022; Rosenfeld et al., 2022) suggests that the true y^* will be much simpler. The maximum disagreement discrepancy here would be approximately 0.5, giving a test accuracy lower bound of 0.5—this is consistent with plausible alternative labeling functions such as y_2^* (orange). Even if y^* is not linear, we still expect that *some* linear function will induce larger discrepancy; this is precisely **??** 3. Suppose instead we learn $\hat{h}$ as depicted in Fig. I.7(c). Then a simple ground truth such as y_3^* (green) is plausible, which would mean $\widehat{h}$ has 0 accuracy on $\mathcal{T}$. In this case, y_3^* is also a critic with disagreement discrepancy equal to 1, and so DIS^2 would correctly output an error upper bound of 1.

I.6.2 DIS^2 on Domain-Adversarial Training Methods

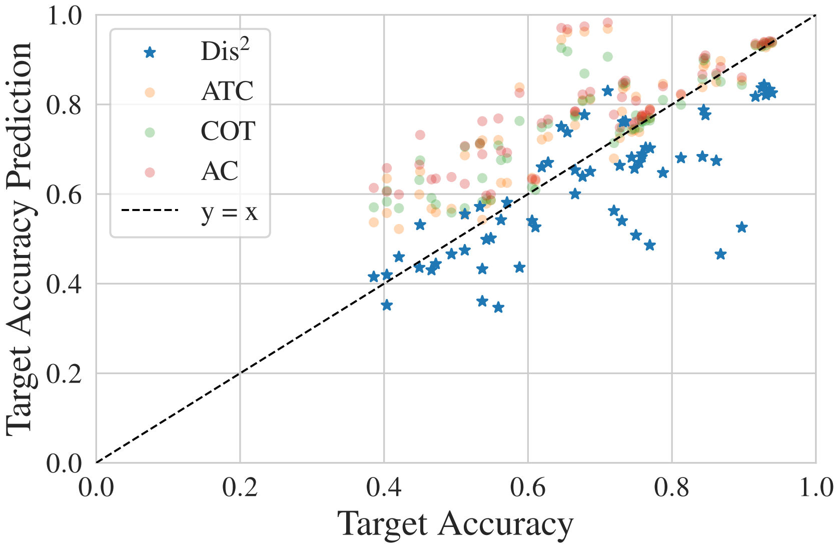


Figure I.8: **DIS^2 may be invalid when the features are explicitly learned to violate Assumption 4.** Domain-adversarial representation learning algorithms such as DANN (Ganin et al., 2016) and CDAN (Long et al., 2018) indirectly minimize $\max_{h' \in \mathcal{H}} \Delta(\widehat{h}, h')$, meaning the necessary condition is less likely to be satisfied. Nevertheless, when DIS^2 does overestimate accuracy, it almost always does so by less than prior methods.

I.7 Proof of Theorem 10.2.4

Proof. Assumption 4 gives $\epsilon_{\mathcal{T}}(\widehat{h}) \leqslant \epsilon_{\mathcal{S}}(\widehat{h}) + \Delta(\widehat{h}, h') = \epsilon_{\mathcal{S}}(\widehat{h}, y^*) + \epsilon_{\mathcal{T}}(\widehat{h}, h') - \epsilon_{\mathcal{S}}(\widehat{h}, h')$. We now define the random variables for $\widehat{\mathcal{S}} \cup \widehat{\mathcal{T}}$:

$$r_i = \begin{cases} {}^{1}\!/_{n_S}, & h'(x_i) = \widehat{h}(x_i) \neq y_i,\ x_i \in \widehat{\mathcal{S}} \\ -{}^{1}\!/_{n_S}, & h'(x_i) \neq \widehat{h}(x_i) = y_i,\ x_i \in \widehat{\mathcal{S}} \\ {}^{1}\!/_{n_T}, & \widehat{h}(x_i) \neq h'(x_i),\ x_i \in \widehat{\mathcal{T}}, \\ 0, & \text{otherwise.} \end{cases}$$

Noting that the expectation of their sum is exactly the above three terms, we apply Hoeffding's inequality: the probability that the expectation exceeds their sum by t is no more than $\exp\left(-\frac{2t^2}{n_S({}^{2}\!/_{n_S})^2 + n_T({}^{1}\!/_{n_T})^2}\right)$. Now simply solve for t. □

Appendix J

Appendix: TiC-CLIP: Continual Training of CLIP Models

J.1 Continual Learning Benchmarks and Methods

We introduce a large-scale image-text benchmark with web scale streaming image text pairs specially developed for studying how efficiently one can get a fresh CLIP model with new incoming batches of data. Table J.1 compares the proposed benchmark with existing datasets for continual learning. Note that this table is not aimed to be an exhaustive list of all CL datasets, but the most popular benchmarks in each domain. For language modeling tasks we report the number of examples/documents as the number of samples and for detection tasks we report the number of labeled objects/bounding boxes.

Table J.1: Comparison with continual learning benchmarks.

Benchmark	# Samples	Years	Time-Continual	Image-Text	Task
Split-MNIST (Goodfellow et al., 2013)	60K	1998	✗	✗	Classification
Perm-MNIST (Goodfellow et al., 2013)	60K	1998	✗	✗	Classification
Rot-MNIST (Lopez-Paz and Ranzato, 2017)	60K	1998	✗	✗	Classification
Split-CIFAR-100 (Zenke et al., 2017)	50K	2008	✗	✗	Classification
Split-MINI-ImageNet (Chaudhry et al., 2019)	50K	2009	✗	✗	Classification
Split-ImageNet (Wen et al., 2020)	1.2M	2009	✗	✗	Classification
Split-ImageNet-R (Wang et al., 2022b)	30K	2019	✗	✗	Classification
CORe50 (Lomonaco and Maltoni, 2017)	165K	2017	✗	✗	Detection
CLAD (Verwimp et al., 2023)	23K	2021	✗	✗	Detection
WANDERLUST (Wang et al., 2021b)	326K	2021	✓	✗	Detection
Inc-PASCAL (Michieli and Zanuttigh, 2019)	11K	2012	✗	✗	Segmentation
Inc-ADE20K (Cermelli et al., 2020)	20K	2012	✗	✗	Segmentation
StreamingQA (Liška et al., 2022)	100K	2007–2020	✓	✗	Question Answering
TemporalWiki (Jang et al., 2022)	32M	2021	✓	✗	Language Modeling
CKL (Jang et al., 2021)	30K	2019-2021	✗	✗	Language Modeling
CTrL (Veniat et al., 2020)	300K	1998-2017	✗	✗	Classification
CLOC (Cai et al., 2021b)	39M	2006-2014	✓	✗	Classification
CLEAR (Lin et al., 2021)	7.8M	2004-2014	✓	✗	Classification
NEVIS (Bornschein et al., 2022)	8M	1992-2021	✓	✗	Classification
Mod-X (Ni et al., 2023)	156K	2014	✗	✓	Retrieval
CLiMB (Srinivasan et al., 2022)	1.3M	2013-2021	✗	✓	Classification
TiC-YFCC	15M	2008-2014	✓	✓	Retrieval / ZS Classification
TiC-RedCaps	12M	2011-2020	✓	✓	Retrieval / ZS Classification
TiC-DataComp	100M/1B/12B	2014-2022	✓	✓	Retrieval / ZS Classification

J.1.1 Extended Related Work

Neural networks trained on new data suffer from catastrophic forgetting of prior knowledge (Goodfellow et al., 2013; Sutton, 1986). Addressing the continual learning challenge, researchers have primarily honed in on methods tailored for small-scale benchmarks, specifically focusing on domain, class, or task incremental benchmarks (Hsu et al., 2018; Van de Ven and Tolias, 2019). Continual learning of foundation models would significantly reduce the costs and increase quick adaptability. While some recent works have started to introduce continual learning benchmarks, they are not naturally time-continual and are comparatively much smaller in scale (Ni et al., 2023; Srinivasan et al., 2022). While evaluations on these benchmarks often neglect the consideration of "training time", it becomes a pivotal factor when scaling continual learning approaches to scenarios involving the training of foundation models such as CLIP.

In our study, we abstain from comparing with continual learning methods that notably prolong the "training time". Methods such as GEM (Chaudhry et al., 2018; Lopez-Paz and Ranzato, 2017), and IMM (Lee et al., 2017), which compute gradients for two models in each training iteration, essentially double the training duration. For completeness, we include a comparison with LWF (Ding et al., 2022; Li and Hoiem, 2017) and EWC (Kirkpatrick et al., 2017). While these methods increase computation cost over standard training due to an additional forward pass, the increase in computation cost is relatively much smaller than methods that compute additional gradients. Our LWF implementation is motivated by Ding et al. (2022) which focuses on continual fine-tuning CLIP models on classification tasks by adapting LwF to CLIP models. Instead, for setups where additional compute resources are available, we run our Cumulative-All approach for slightly longer. Cumulative-All narrows the gap with Oracle (refer to Table 11.2). Given that data storage costs are substantially lower than computational costs at scale, we advocate for taking computational efficiency into consideration in future endeavors.

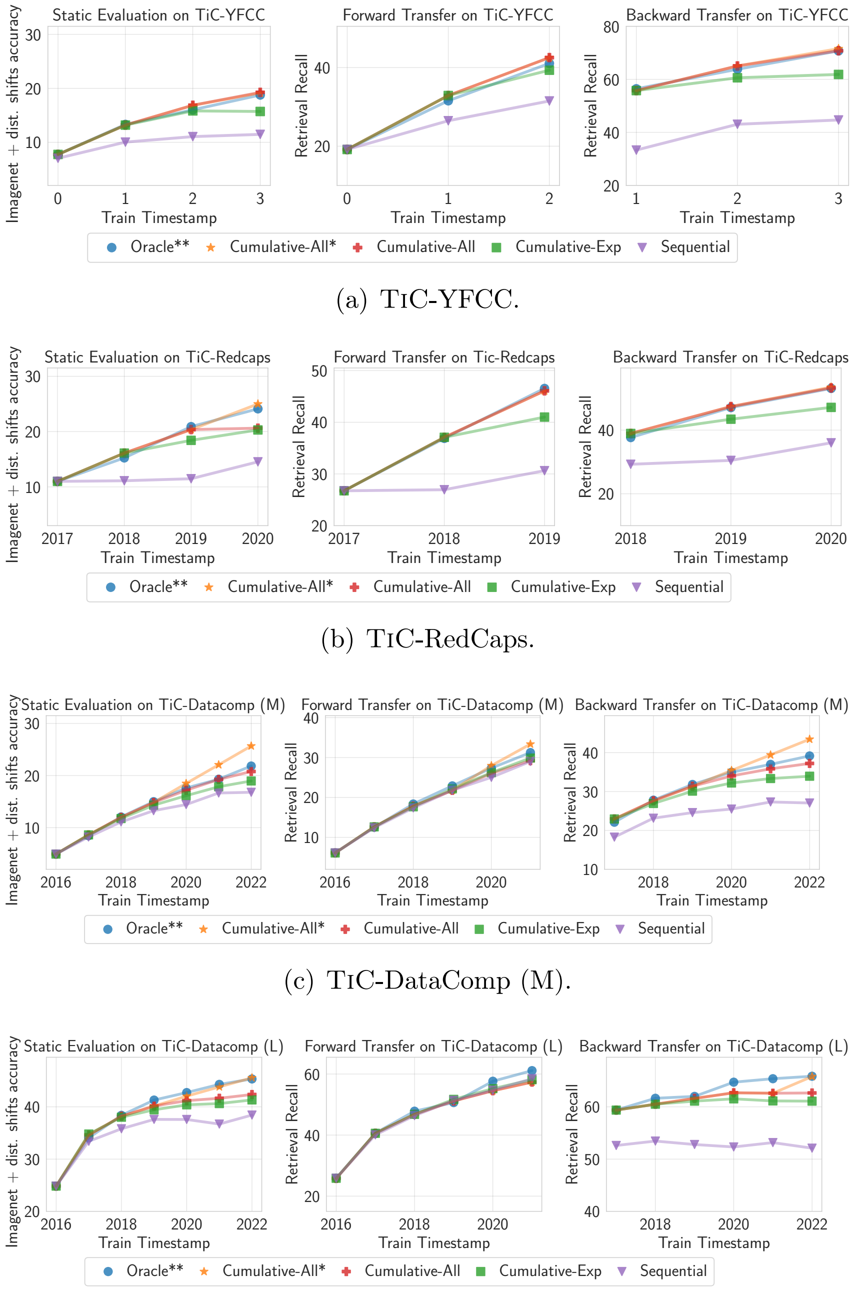
J.1.2 Discussion and comparison with CLOC Benchmark

Cai et al. (2021b) provide interesting discussion/analysis for continual learning at a large number of steps. However, our study differs from Cai et al. (2021b) in several crucial respects: (i) Training Methodology: We employ noisy supervision using contrastive loss between image-text pairs, as opposed to the cross-entropy loss used by Cai et al. (2021b). (ii) Scale of Experiments: Our experiments on the TiC-DataComp dataset are orders of magnitude larger, scaling up by 200×.

These differences introduce unique challenges. The use of contrastive loss (i) necessitates a tailored approach to designing our evaluation studies. The significantly larger scale of our experiments (ii) poses challenges in collecting timestamped data and understanding if and how distribution shifts impact learning at this scale.

J.2 Additional Experimental Results

J.2.1 Detailed Results on Our Benchmarks



(a) TIC-YFCC.

(b) TIC-RedCaps.

(c) TIC-DataComp (M).

(d) TIC-DataComp (L).

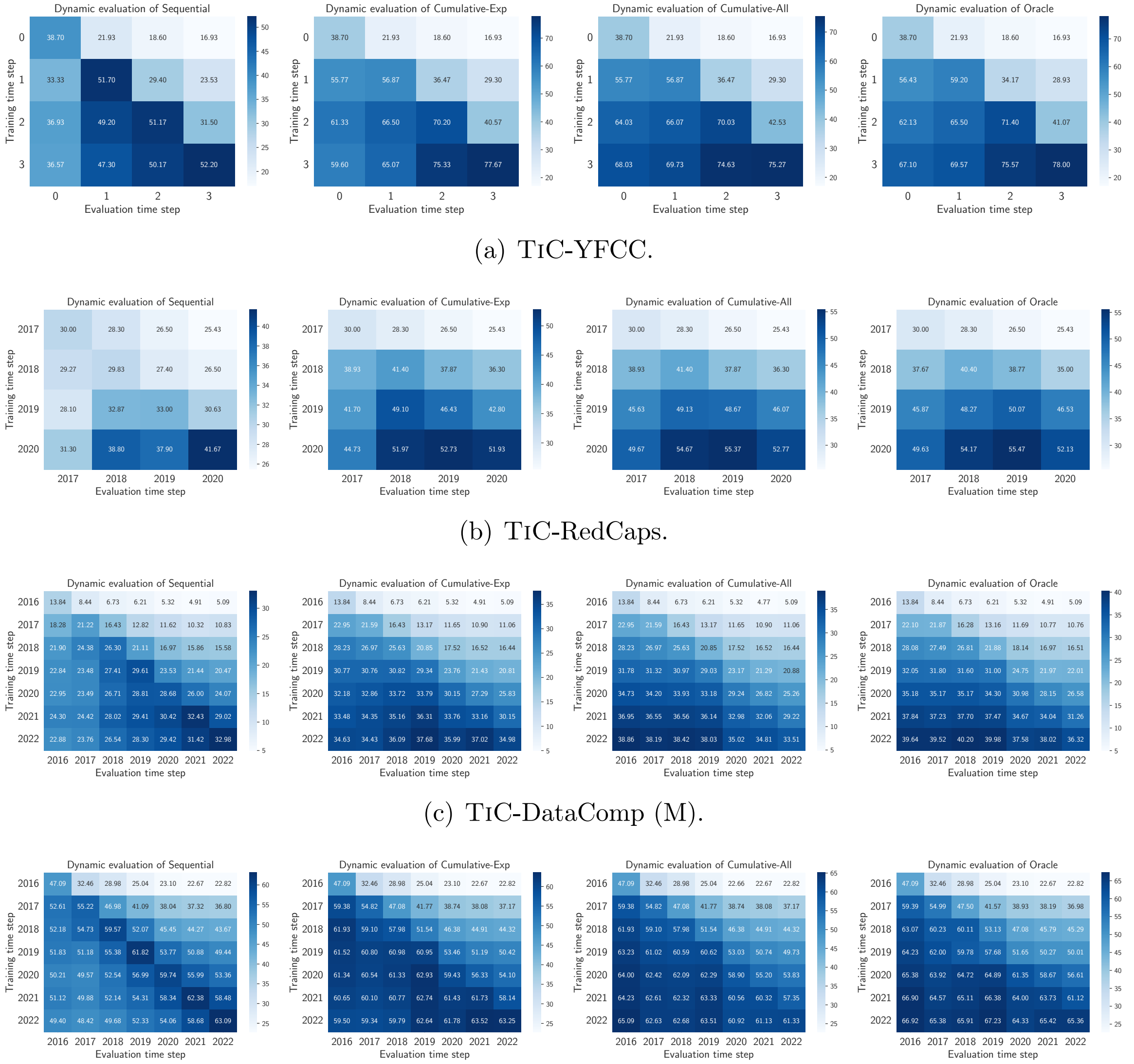
Figure J.1: **Static and dynamic evaluation performance over time with selected methods in our testbed.** As we get more data, all methods improve on both static and forward transfer on dynamic tasks but methods with limited replay buffer start performing slightly worse for backward transfer.

J.2.2 Results with Basic Filtering on TIC-DataComp XL

Filtering strategy changes the ordering of performance on static and dynamic retrieval tasks. We observe that while Bestpool filtering models outperform basic filterining models on TIC-DataComp (XL) by 6% on static tasks, they underperform by over 5% on dynamic retrieval task (see Fig. J.3). In the main paper (Table 11.2), we included TIC-DataComp (`xlarge`) results with Bestpool filtering. In Table J.2, we include basic filtering results. We observe that while Bestpool filtering models perform better than basic filtering models on static tasks, the order is flipped on dynamic retrieval tasks. Hence, we resort to including results with Basic filtering at smaller scales, but include Bestpool results for completeness as it achieves better results on static tasks.

Table J.2: **Zero shot performance on our time-continual benchmarks (Basic and Bestpool filtering).** * and ** denote methods that violate the compute budget and use extra compute. For static tasks, we tabulate accuracy of the models obtained on the final timestamp. For dynamic tasks, we tabulate forward transfer, backward transfer and ID performance. For all metrics, higher is better. Bestpool filtering results are copied from Table 11.2.

Benchmark	**Method**	**Compute** (MACs)	**Static Tasks**				**Dynamic Retrieval Tasks**		
			ImageNet	ImageNet dist. shift	Flickr30k	Average over 28 datasets	Backward Transfer	ID Performance	Forward Transfer
TIC-DataComp	Sequential	2.7×10^{20}	66.5	54.2	61.2	61.0	63.1	68.9	56.8
	Cumulative-All	2.7×10^{20}	71.6	58.8	65.1	64.8	**70.7**	**68.5**	**57.1**
	Cumulative-All*	3.5×10^{20}	**72.8**	60.4	66.5	**66.7**	**71.0**	**68.6**	**57.1**
	Oracle**	1.1×10^{21}	**73.3**	**61.3**	**68.0**	**65.8**	-	-	-
TIC-DataComp	Cumulative-All	2.7×10^{20}	63.5	52.0	62.8	58.7	64.6	55.5	47.6
	Sequential	2.7×10^{20}	60.2	48.9	62.4	56.6	51.6	50.3	45.0
	Oracle**	1.1×10^{21}	66.0	54.0	63.8	59.6	-	-	-



(a) TɪC-YFCC.

(b) TɪC-RedCaps.

(c) TɪC-DataComp (M).

(d) TɪC-DataComp (L).

Figure J.2: Dynamic retrieval evaluation results on our benchmarks with Sequential, Cumulative-Exp, Cumulative-All and Oracle. These evaluations highlight the catastrophic forgetting observed with Sequential and Cumulative-Exp. Moreover, by observing new data, we not only benefit on tasks from current time step but also improve performance on tasks from old time steps.

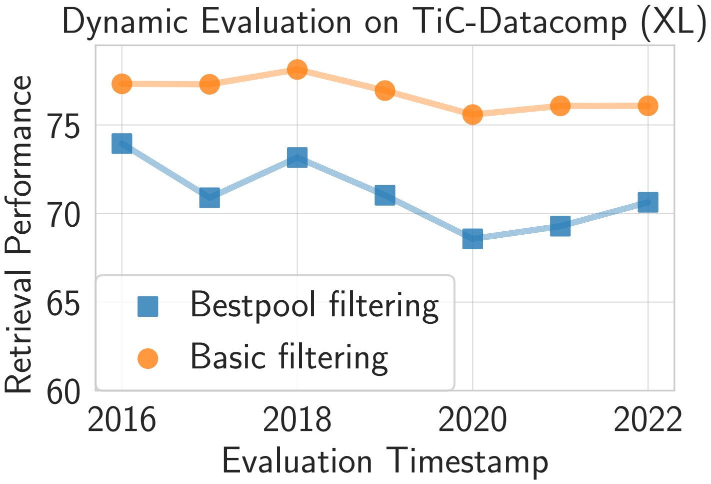


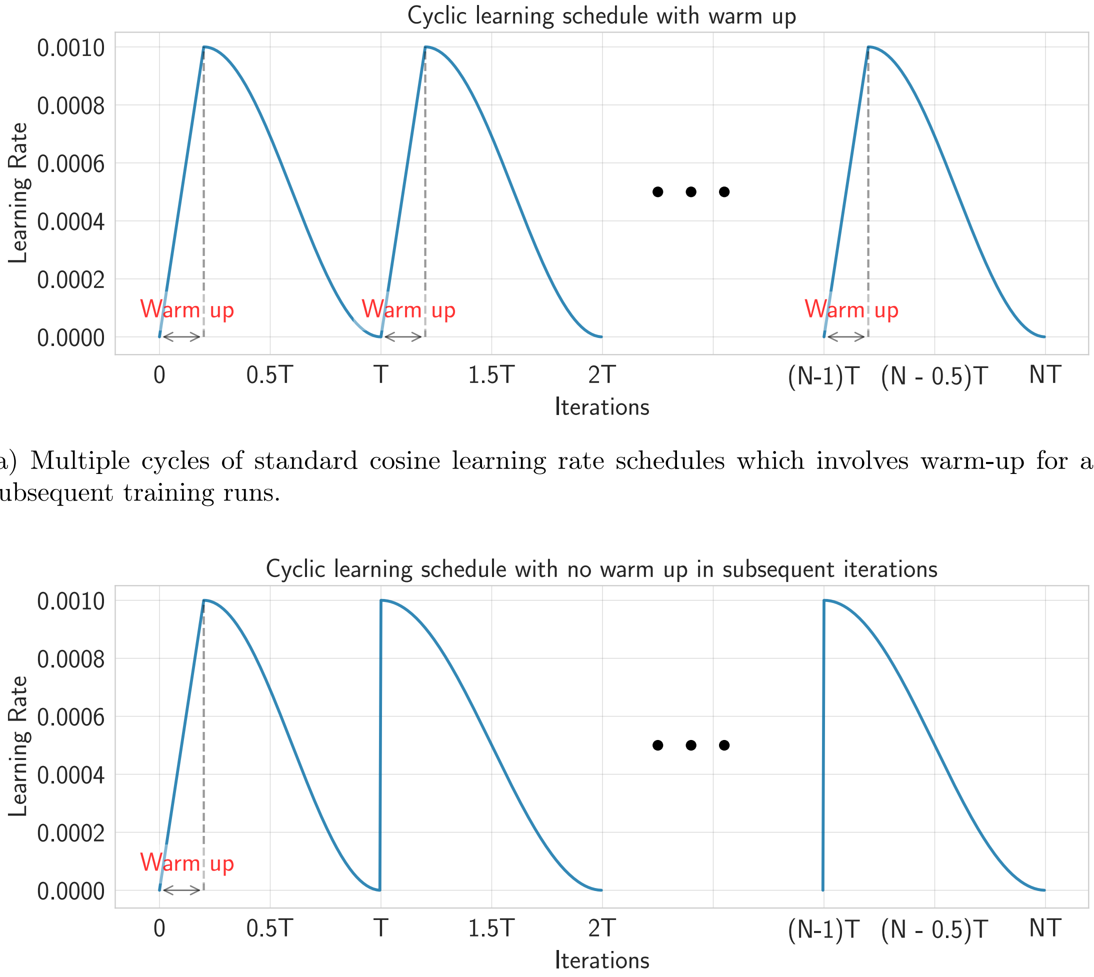
Figure J.3: Comparing Oracle models trained on Bestpool and Basic filtering trained on data from all time steps. Our results clearly highlight that Basic filtering performs better than Bestpool filtering on dynamic retrieval task. However, on static tasks, the order is reversed. Moreover, Bestpool filtering shows a drop in retrieval performance from 2016 to 2022 when compared with Basic filtering.

J.2.3 Ablations with learning rate warmup and maximum learning rate

To continually train models as more data arrives sequentially over time, we use multiple cycles of cosine learning rate schedule (Fig. J.4). There are two crucial design choices: (i) Should we warm up the learning rate for subsequent continual runs? and (ii) How should the maximum learning rate change for sequential training runs?

Table J.3: **Zero shot performance on our time-continual benchmarks with and without initial LR wamrup for subsequent runs.** Using warm up on sequential runs after training on the first time step hurts slightly when compared with not using warm up on sequential runs.

Benchmark	**Method**	**Static Tasks**				**Dynamic Retrieval Tasks**		
		ImageNet	ImageNet dist. shift	Flickr30k	Average over 28 datasets	Backward Transfer	ID Performance	Forward Transfer
TiC-DataComp (M)	Cumulative-All (w/o warmup)	24.0	20.2	20.9	17.9	33.8	26.4	15.1
	Cumulative-All (w warmup)	23.3	20.1	20.3	17.6	33.3	26.1	14.8
TiC-DataComp (L)	Cumulative-All (w/o warmup)	48.9	41.3	50.9	36.3	62.1	57.3	41.2
	Cumulative-All (w warmup)	47.6	40.6	50.0	35.2	60.1	53.0	39.5



(a) Multiple cycles of standard cosine learning rate schedules which involves warm-up for all subsequent training runs.

(b) Our proposed cosine learning rate schedule without learning rate warm-up for subsequent training runs.

Figure J.4: **Learning rate schedule ablations.** Schedules vary on how continual training is performed when the training run is initialized with the best previous model. When training with cosine learning schedules for subsequent runs, we observe that keeping the same maximum learning rate as the first run performs the best.

Table J.4: Cumulative experiments on TiC-DataComp (M) with different maximum learning rates for subsequent runs with first run fixed at LR 0.00025. Our default choice for subsequent runs is 0.00025. Performance reported on ImageNet. At maximum learning rate 0.001, the runs crashed with Nan in loss.

Method	**Max LR**				
	0.00005	0.0001	0.00025	0.0005	0.001
Cumulative-All	16.3	19.0	24.0	10.1	–

When training with large batches, linear learning rate warm-up is typically employed to stabilize the start of the training when beginning from a random initialization (Goyal et al., 2017; Steiner et al., 2021). However, when training sequentially by initializing models with checkpoints from the previous step, it remains unclear whether we should employ a learning rate warm up or not. Our observations highlight that while warm up is benefits for the first time step, not using warm up on subsequent runs performs better. In particular, we observe that removing the warm up for the first training run hurts the final performance. On TIC-DataComp (`large`), we observe that training a ViT-B/16 with warm up on the first time step (i.e., 2016) gets 29.9 zero-shot on Imagenet, whereas, without warm up ViT-B/16 achieves only 24.1 zero-shot performance on Imagenet. Table J.3 shows the final performance of models trained with and without warmup on subsequent time steps (after training on the first time step with warmup). In particular, on TIC-DataComp (`large`), we observe 1.5% accuracy gap on Imagenet and 4.3% accuracy gap on dynamic ID retrieval performance on models trained with and without warm up.

Hence, we default to using warmup when training on the first time step and not using it on the subsequent time steps with all methods except for training on TIC-DataComp (XL) where we add a smaller warm up (10% of the warm up iterations used in first step) to stabilize training.

Next, we experiment with different maximum learning rate when training with cosine schedules. We ablate on TIC-DataComp (M) to investigate how to change LR after training on data from the first time step. Unlike conventional pretraining and finetuning settings where LR is typically decreased for subsequent training, we observe that decaying maximum LR for subsequent steps in our setup hurts on static and dynamic tasks and consequently, we use the same maximum LR across our runs (see Table J.4).

J.2.4 Preliminary experiments comparing random subsampling with other strategies to reduce buffer size

In our preliminary experiments, we explored the efficacy of subsampling old data based on the alignment between text and image content from previous time steps. Specifically, when training a model at time step $t + 1$, we used the model from the end of time step t to assess this alignment. We employed two distinct subsampling methods:

1. Retaining half of the data with the lowest alignment scores, based on the premise that these data points might be more challenging to learn and require additional gradient steps.

2. Retaining half of the data with the highest alignment scores, under the assumption that these represent higher quality data, as indicated by the stronger alignment between text and image pairs.

We applied these methods to the TiC-YFCC dataset and evaluated their performance against a baseline of random sampling. The outcomes revealed minimal differences: less than 0.2% variation in Imagenet performance and under 0.5% in dynamic retrieval performance across different time steps. Given that these minor improvements came with a significant computational cost—requiring a full forward pass to compute alignment post each training epoch—they exceeded our compute budget constraints. As a result, we opted for random sampling in our research. We leave investigation on improved subsampling techniques for future work.

J.2.5 Const-Cosine: An alternative learning rate schedule

The defacto LR schedule for training CLIP models is an initial linear increase to a maximum value, i.e., warm up, followed by a cosine decay (Gadre et al., 2023; Radford et al., 2021). In the main paper, we default to using cosine LR schedule for each sequential run, resulting in a cyclic schedule. We observe a significant increase in training loss early in subsequent runs when the LR is high. Comparing the loss on training data with Cumulative and Oracle methods, we observe that as training progresses the training loss increases every time the learning rate is increased to the maximum LR (Fig. J.5).

It would be ideal for continual training to employ a learning rate schedule that is "forward looking", allowing us to continually train from a previous checkpoint without experiencing a significant increase in training loss. One desirable property of such a learning rate schedule would be its ability to adapt without requiring prior knowledge of the decay period.

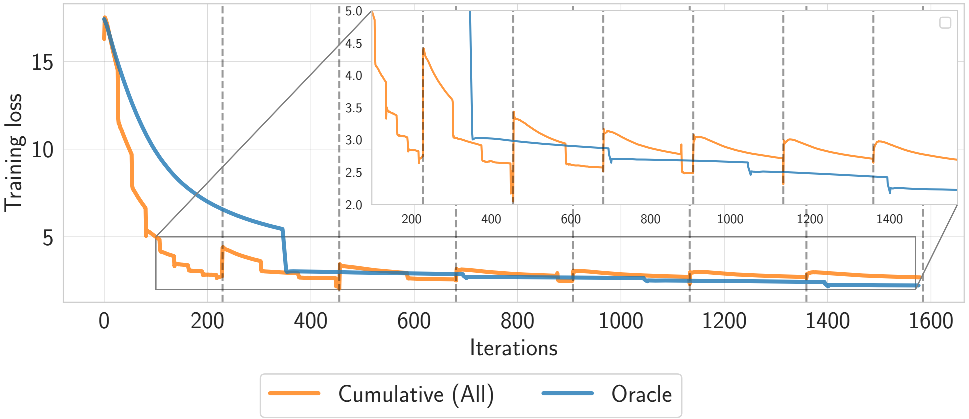


Figure J.5: **Training loss increases every time the LR is reset to maximum LR for Cumulative.** Loss comparison on training data with Cumulative and Oracle method. Cumulative is trained with a cyclic cosine schedule without warm up for sequential training runs. For Cumulative, we plot the loss on training data, and as the training progresses, samples from new time steps are added to the training pool. For Oracle, the training data is the union of data from all time steps and remains the same throughout the training.

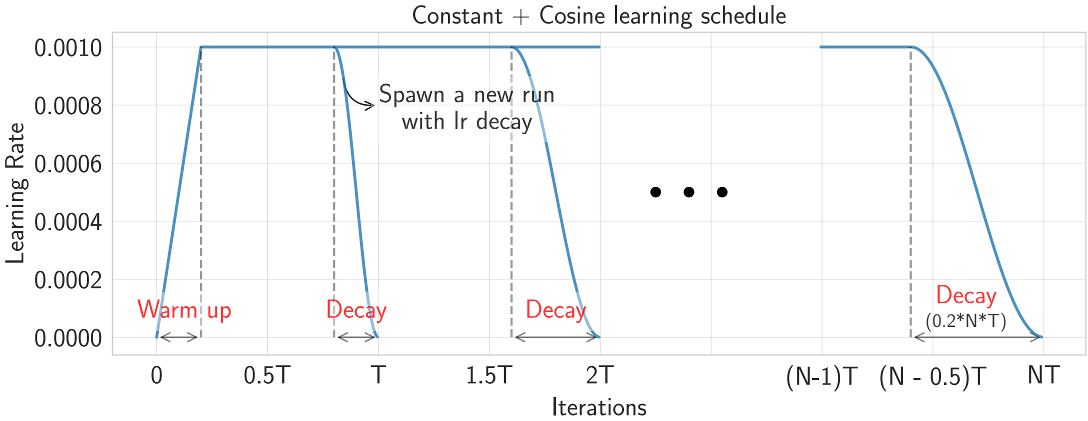


Figure J.6: Const-Cosine: Our proposed alternative forward-looking learning rate schedule schedule which trains one model with constant learning rate and decays the learning rate with cosine schedule only for a fraction of iterations before obtaining a deployable model. Const-Cosine schedule uses an extra compute budget than an Oracle run because an extra training run is launched for the fraction of training when learning rate is decayed.

In our work, we perform preliminary experiments with the *simplest* alternative, Const-Cosine where after the warm up period, we train with a constant learning rate and decay the learning rate only for a small fraction of training towards the end when we want a deployable model (Fig. J.6). This allows us to continue training for subsequent runs from the checkpoint at the end of the constant learning rate schedule and decay the LR only in the end. For our experiments, we fix the decay period as 0.2 of the total training iterations. Due to this, Const-Cosine schedule slightly increases the overall training budget of the Cumulative runs when compared with cyclic cosine schedules.

For Const-Cosine, we only ablate at relatively smaller scale datasets in our testbed (i.e., TIC-YFCC, TIC-RedCaps, and TIC-DataComp (medium)). For a fair comparison, we also re-run Oracle methods with the same Const-Cosine schedule. Note that for Const-Cosine experiments, we use the same maximum LR as with the cosine schedule.

We observe that training with Const-Cosine schedule significantly improves both Cumulative and Oracle as compared to their counterparts trained with cosine learning rates [1]. Moreover, as expected, we do not observe jumps in training loss when training Cumulative with Const-Cosine schedule. However, the gap between Oracle and Cumulative with Const-Cosine doesn't decrease when compared with gap between Oracle and Cumulative with cosine learning rate schedules. This highlights that the jumps in the training loss observed

[1] We also experimented with Const-Cosine schedule for Oracle training on TIC-DataComp (large) and TIC-DataComp (xlarge). We observe that with a decay fraction of 0.2, Const-Cosine achieves similar results to that of the cosine learning rate schedule. In particular, Const-Cosine achieves 61.3 on large and 73.0 on xlarge versus Cosine schedule achieves 62.3 on large and 73.3 on xlarge. This highlights the potential of training with Const-Cosine schedule in scenarios where total training duration might be unknown apriori.

while training with the cyclic cosine schedule might have benign effects on the final performance.

Table J.5: **Zero shot performance on Imagenet with Const-Cosine LR schedule.** We observe that Const-Cosine improves over cyclic cosine LR schedule. However, the gap between cyclic cosine LR schedule and Const-Cosine for different LR schedules remains the same. ** denote methods that violate the compute budget.

	Method	**Cosine LR Schedule**		**Const-Cosine LR schedule**	
Benchmark		Compute (MACs)	ImageNet	Compute (MACs)	ImageNet
TiC-YFCC	Cumulative-All	3.4×10^{18}	29.3	4.4×10^{18}	32.8
	Oracle**	8.5×10^{18}	29.2	8.5×10^{18}	33.2
TiC-RedCaps	Cumulative-All	3.4×10^{18}	32.2	4.4×10^{18}	35.1
	Oracle**	8.5×10^{18}	32.7	8.5×10^{18}	36.2
TiC-DataComp (M)	Cumulative-All	3.0×10^{18}	24.0	3.6×10^{18}	28.2
	Oracle**	1.2×10^{19}	25.5	1.2×10^{19}	28.9

J.2.6 OpenCLIP models obtained by retraining after removing any duplicate examples from the test set

OpenCLIP models (e.g., models trained on Datacomp and LAION-5B) have been trained on data curated from Common Crawl. Since the retrieval tasks we constructed are built on top of data curated from Common Crawl, one may argue there is a possibility of train/test overlap in our evaluations of OpenCLIP models. Thus, we retrain OpenCLIP models on DataComp datasets after removing the samples in our test sets. Figure J.7 shows that the trends observed for OpenCLIP models holds for our retrained models.

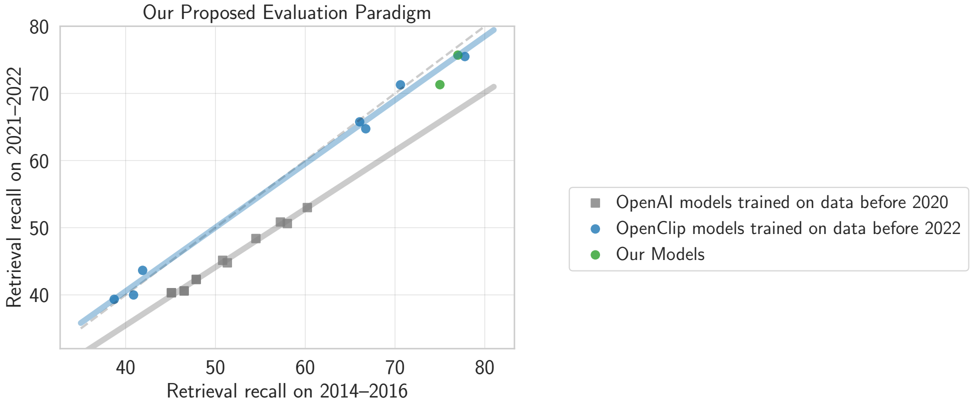


Figure J.7: We replicate OpenCLIP models by training from scratch and removing duplicates from the evaluation dataset. We observe that trends continue to hold.

J.2.7 Results on dynamic classification task

In the main paper, we include results on our dynamic retrieval task. For completeness, here we include results on dynamic classification tasks on TiC-DataComp splits (Table J.6).

Along with including results on all nodes of ImageNet, we also include results on classification task restricted to classes in the "motor vehicles" subtree of ImageNet hierarchy. For the dynamic classification task, we observe trends similar to the dynamic retrieval task.

Table J.6: **Zero shot performance on our TiC-DataComp-Net classification task.** * and ** denote methods that violate the compute budget. We tabulate forward/backward transfer and ID performance on classification tasks (Sec. 11.2.3). For TiC-DataComp (XL), we include results with Bestpool filtering.

Benchmark	**Method**	**Compute** (MACs)	**Dynamic Retrieval Tasks (All)**			**Dynamic Retrieval Tasks ('Motor Vehicles')**		
			Backward Transfer	ID Performance	Forward Transfer	Backward Transfer	ID Performance	Forward Transfer
TiC-DataComp (M)	Sequential	3.0×10^{18}	15.9	13.3	9.9	34.5	30.0	22.6
	Patching	3.0×10^{18}	15.6	13.1	9.7	34.4	29.2	22.1
	Cumulative-Exp	3.0×10^{18}	17.6	14.4	10.4	36.6	30.9	23.5
	Cumulative-Equal	3.0×10^{18}	17.5	14.2	10.4	36.4	31.1	23.5
	Cumulative-All	3.0×10^{18}	18.3	14.7	10.6	38.2	31.7	23.7
	LwF*	3.8×10^{18}	16.0	13.5	9.9	35.1	30.7	23.3
	Cumulative-All*	3.9×10^{18}	20.7	16.0	10.9	40.4	32.3	23.9
	Oracle**	1.2×10^{19}	19.2	15.2	10.7	38.7	31.9	23.5
TiC-DataComp (L)	Sequential	2.7×10^{19}	38.3	36.9	33.3	58.4	55.6	49.7
	Patching	2.7×10^{19}	38.6	36.8	33.3	58.3	54.9	49.3
	Cumulative-Exp	2.7×10^{19}	40.2	37.9	34.2	60.7	56.8	51.1
	Cumulative-Equal	2.7×10^{19}	40.6	38.0	34.2	60.7	56.8	50.8
	Cumulative-All	2.7×10^{19}	41.3	38.3	34.4	61.4	56.6	50.9
	Cumulative-All*	4.1×10^{19}	43.0	39.2	34.6	62.7	57.5	51.1
	Oracle**	1.1×10^{20}	43.8	40.0	35.2	62.6	56.8	50.7
TiC-DataComp (XL)	Sequential	2.7×10^{20}	55.4	55.1	53.3	67.8	66.0	63.5
	Cumulative-All	2.7×10^{20}	58.5	56.7	54.3	70.2	67.4	63.8
	Cumulative-All*	3.5×10^{20}	58.8	56.9	54.3	70.5	67.5	63.8

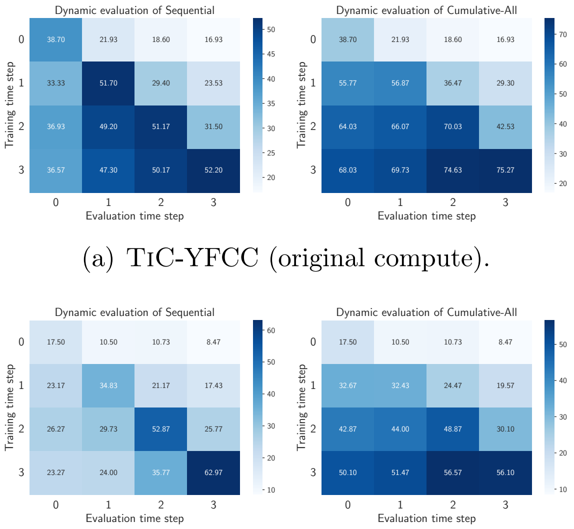
J.2.8 Addressing differences between Sequential and Cumulative-All between TiC-YFCC and TiC-DataComp

In Table 11.2, we observe differences in the behavior of Sequential and Cumulative-Allon TiC-YFCC when compared with TiC-DataComp. For instance, differences between the ID performance between Sequential and Cumulative-All is larger in TiC-YFCC than in TiC-DataComp (M). Similar observations hold true for backward transfer performance. In this section, we explain the underlying causes for these differences.

We identify two primary reasons:

(i) the nature of the distribution shift observed in TiC-YFCC. We observe that models trained with Sequential on TiC-YFCC suffer from relatively larger drops on old-time steps than TiC-DataComp (M) due to catastrophic forgetting (see Fig. J.2).

(ii) compute used at each time step per data available at each time step is different for these bencmarks. Overall YFCC is 2x smaller than Tic-Datacomp (M) but the compute we used in both TiC-YFCC and TiC-Datacomp setup is of similar order (in fact, it is slightly higher in TiC-YFCC). We re-ran the experiments for Tic-YFCC by reducing the compute. In the updated runs, we observe that the gap between ID performances of Sequential and Cumulative-All vanishes.



(a) TIC-YFCC (original compute).

(b) TIC-YFCC (reduced compute).

Figure J.8: Dynamic retrieval evaluation results with Sequential, Cumulative-All on TIC-YFCC with reduced compute.

Table J.7: **Zero shot retrieval performance on TIC-YFCC with Sequential and Cumulative-All with reduced compute.**

Benchmark	**Method**	**Dynamic Retrieval Tasks (original compute)**				**Dynamic Retrieval Tasks (reduced compute)**			
		Compute (MACs)	Backward Transfer	ID Performance	Forward Transfer	**Compute** (MACs)	Backward Transfer	ID Performance	Forward Transfer
TIC-YFCC	Sequential	3.4×10^{18}	42.2	48.4	23.7	1.5×10^{18}	27.0	42.0	15.7
	Cumulative-All	3.4×10^{18}	66.4	60.2	27.6	1.5×10^{18}	46.3	38.7	17.3

J.3 Additional Benchmark Details

J.3.1 Filtering ablations on TIC-DataComp

For Basic Filtering, Gadre et al. (2023) performs the following three steps: filter by English language (using fasttext (Joulin et al., 2017)), filter by caption length over two words and 5 characters, and filter by image sizes with smallest dimensions over 200 pixels and aspect ratio above 3. We do not default to other filtering techniques that use off-the-shelf CLIP models from Gadre et al. (2023) to avoid biasing dataset selection from each time step. In Fig. J.9, we show that "Bestpool" filtering (which filters image-text pairs with CLIP scores and ImageNet image embeddings) biases dataset selection to preferring old time step data over new timestamp data. Moreover, we also show that models trained with Bestpool filtering is less robust when evaluated on our dynamic tasks from 2021-2022 (Fig. J.9). Nevertheless, for completeness and to highlight the significance of our findings even for state-of-the-art filtering techniques, we perform continual learning experiments with Bestpool filtering at `xlarge` scale which is included in the main paper. In App. J.2.2, we include results with Basic filtering at `xlarge`.

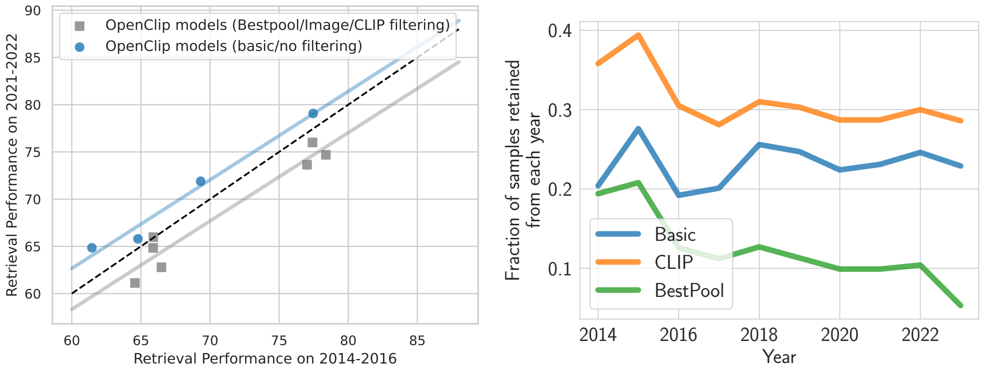


Figure J.9: (Left) Gap in retrieval performance for different OpenCLIP models that use different filtering techniques. (Right) Reduction in TIC-DataComp data at different times with different filtering techniques. This clearly highlights that there is a selection bias towards retaining more old data for CLIP/BestPool filtering. No such bias exists for basic filtering.

J.3.2 Static Datasets considered for evaluation

Table J.8: Evaluation tasks borrowed from Gadre et al. (2023).

Task type	Dataset	Task	Test set size	Number of classes	Main metric
	Food-101 (Bossard et al., 2014)	Food recognition	25,250	101	accuracy
	GTSRB (Stallkamp et al., 2011)	Traffic sign recognition	12,630	43	accuracy
	ImageNet 1k (Deng et al., 2009)	Visual recognition	50,000	1,000	accuracy
	ImageNet Sketch (Wang et al., 2019a)	Visual recognition	50,889	1,000	accuracy
	ImageNet V2 (Recht et al., 2019a)	Visual recognition	10,000	1,000	accuracy
	ImageNet-A (Hendrycks et al., 2021c)	Visual recognition	7,500	200	accuracy
	ImageNet-O (Hendrycks et al., 2021c)	Visual recognition	2,000	200	accuracy
	ImageNet-R (Hendrycks et al., 2021a)	Visual recognition	30,000	200	accuracy
	KITTI distance (Geiger et al., 2012; Zhai et al., 2019)	Distance prediction	711	4	accuracy
	MNIST (LeCun et al., 1998)	Digit recognition	10,000	10	accuracy
	ObjectNet (Barbu et al., 2019)	Visual recognition	18,574	113	accuracy
Classification	Oxford Flowers-102 (Nilsback and Zisserman, 2008)	Flower recognition	6,149	102	mean per class
	Oxford-IIIT Pet (Parkhi et al., 2012; Zhai et al., 2019)	Pet classification	3,669	37	mean per class
	Pascal VOC 2007 (Everingham et al., 2007)	Object recognition	14,976	20	accuracy
	PatchCamelyon (Veeling et al., 2018; Zhai et al., 2019)	Metastatic tissue cls.	32,768	2	accuracy
	Rendered SST2 (Zhai et al., 2019)	Sentiment classification	1,821	2	accuracy
	RESISC45 (Cheng et al., 2017; Zhai et al., 2019)	Satellite imagery recognition	6,300	45	accuracy
	Stanford Cars (Krause et al., 2013)	Vehicle recognition	8,041	196	accuracy
	STL-10 (Coates et al., 2011)	Visual recognition	8,000	10	accuracy
	SUN-397 (Xiao et al., 2016)	Scene recognition	108,754	397	accuracy
	SVHN (Netzer et al., 2011b; Zhai et al., 2019)	Digit recognition	26032	10	accuracy
	iWildCam (Beery et al., 2020; Koh et al., 2021)	Animal recognition	42,791	182	macro F1 score
	Camelyon17 (Bandi et al., 2018; Koh et al., 2021)	Metastatic tissue cls.	85,054	2	accuracy
	FMoW (Christie et al., 2018; Koh et al., 2021)	Satellite imagery recognition	22,108	62	worst-region acc.
Retrieval	Flickr30k (Young et al., 2014)	Image and text retrieval	31,014	N/A	R@1

J.3.3 Our Benchmark Statistics

In this section, we discuss statistics of our constructed benchmarks. Fig. J.10 summarizes TɪC-RedCaps, TɪC-YFCC and TɪC-DataComp dataset sizes. Fig. J.11 summarizes original YFCC dataset sizes. Table J.9, Table J.10 and Table J.11 present the exact numbers for these datasets. For TɪC-DataComp, we only discuss the sizes at `xlarge` scale.

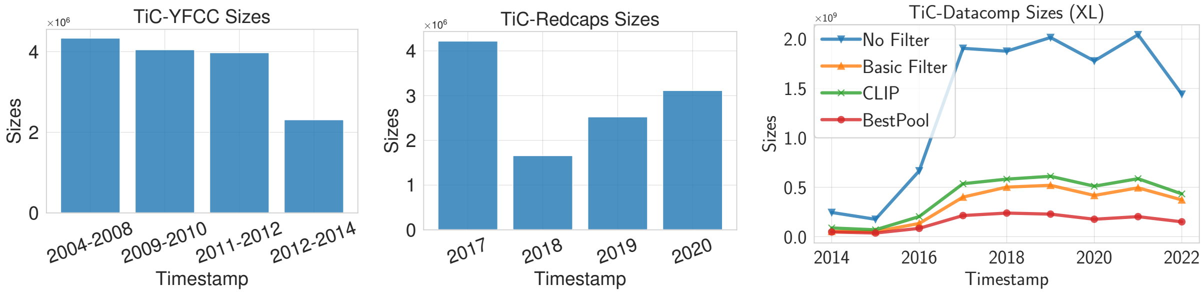


Figure J.10: Number of examples in each year in our benchmarks.

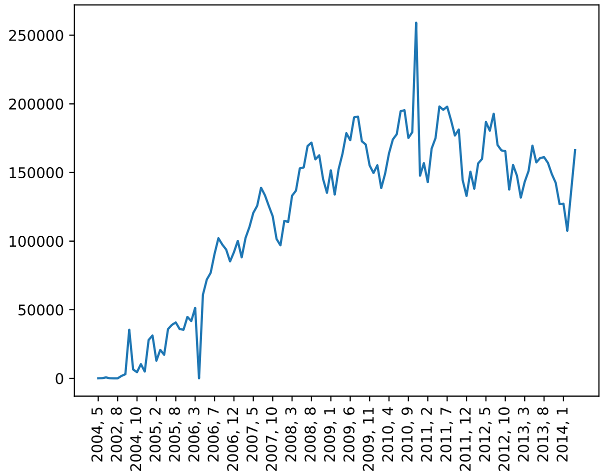


Figure J.11: Number of examples in each year in original YFCC 15M. X-axis the upload month and y-axis is the number of examples in that month.

Table J.9: Number of examples in TɪC-RedCaps in each year.

Dataset	**Year**			
	2017	2018	2019	2020
TɪC-RedCaps	4,220,262	1,660,003	2,526,575	3,115,715

Table J.10: Number of examples in TɪC-YFCC in each year.

Dataset	**Year**			
	2004–2008	2009–2010	2011–2012	2012–2014
TɪC-YFCC	4,337,727	4,050,166	3,976,339	2,312,753

Table J.11: Number of examples in TiC-DataComp in each year before filtering.

Dataset	**Year**								
	2014	2015	2016	2017	2018	2019	2020	2021	2022
TiC-DataComp (no filter)	244,802,598	175,648,045	666,019,511	1,906,357,755	1,877,561,875	2,016,011,588	1,778,751,066	2,044,463,701	1,442,233,121
TiC-DataComp (basic filter)	52,764,775	50,757,898	133,333,267	400,225,598	501,347,511	519,575,760	417,067,014	494,038,122	371,748,613

Next, we tabulate the number of examples in our retrieval evaluation datasets. Since the evaluation dataset sizes are different at different time steps, we subsample the dataset to a fixed size before performing retrieval evaluations. On TiC-YFCC and TiC-RedCaps, we randomly sampled 1000 image-text pairs from these evaluation datasets. For TiC-DataComp, we randomly sample 4000 image-text pairs. We repeat this process for 3 seeds and report the aggregated performance.

Table J.12: Number of retrieval evaluation examples in TiC-RedCaps in each year.

Dataset	**Year**			
	2017	2018	2019	2020
TiC-RedCaps	31,316	42,539	16,738	25,565

Table J.13: Number of retrieval evaluation examples in TiC-YFCC in each year.

Dataset	**Year**			
	2004–2008	2009–2010	2011–2012	2012–2014
TiC-YFCC	43,820	40,909	40,165	23,354

Table J.14: Number of retrieval evaluation examples in TiC-DataComp in each year before filtering.

Dataset	**Year**						
	2016	2017	2018	2019	2020	2021	2022
TiC-DataComp	23,085	39,289	50,450	53058	42,239	49,841	38,051

J.3.4 Compute Constraints for Different Datasets

We closely follow compute budget constraints from Gadre et al. (2023). In particular, on TiC-DataComp, we restrict to using exactly the same amount of overall compute as fixed in Gadre et al. (2023). Below we list exact total MACs on each dataset:

- TiC-YFCC: Total MACs: 3.4×10^{18}
- TiC-RedCaps: Total MACs: 3.4×10^{18}
- TiC-DataComp `medium`: Total MACs: 3.0×10^{18}
- TiC-DataComp `large`: Total MACs: 2.7×10^{19}
- TiC-DataComp `xlarge`: Total MACs: 2.7×10^{20}

For a ViT-B architecure, these values correspond to 20k iterations on TIC-YFCC (batch size: 8192), TIC-RedCaps (batch size: 8192), 35k iterations on TIC-DataComp (M) (batch size: 4096), 157k iterations on TIC-DataComp (L) (batch size: 8192), and 143.5k iterations on TIC-DataComp (XL) (batch size: 90100). We divide these iterations equally among all time steps.

J.3.5 Creation Pipeline for Evaluation Datasets

TIC-DataComp-Retrieval To create a retrieval task, we sample a batch of IID image-text pairs from different timestamps and evaluate text retrieval performance given the corresponding image (similarly, image retrieval given the corresponding text). Alongside general evaluations, we also construct datasets from specific domains, e.g., Covid-19 subset and Flickr subset. To create Covid-19, we filter the dataset to only retain pairs where the caption contains a mention of "covid". This search process restricts the data to time only after 2019. For the Flickr subset, we filter the dataset to only retain pairs where the corresponding "url" contains data from Flickr.

TIC-DataComp-Net We create our dynamic classification dataset TIC-DataComp-Net with ImageNet classes from the CommonPool data augmented with temporal information. Our construction process draws inspiration from the LAIONet construction process described in Shirali and Hardt (2023). In particular, we first filter examples where the corresponding caption contains one and only one of the synsets of ImageNet-1K. We also apply additional basic filtering (Gadre et al., 2023) to make sure that images are of at least 200 size in smallest dimension and the caption contains at least 2 words and 5 characters. After filtering for examples with ImageNet synsets, we only retain examples where the similarity—as evaluated by an off-the-shelf sentence embedding model (Reimers and Gurevych, 2019)—between imagenet synset definition and the caption exceeds a threshold of 0.5. The goal of this filtering step is to restrict examples with "high" alignment between caption and imagenet synset definition. This last step differs from the LAIONet construction. Crucially, unlike LAIONet, we do not filter the image-text pairs with CLIP similarity scores to avoid biasing the dataset selection process.

J.3.6 Distribution Shift Analysis on Proposed benchmarks

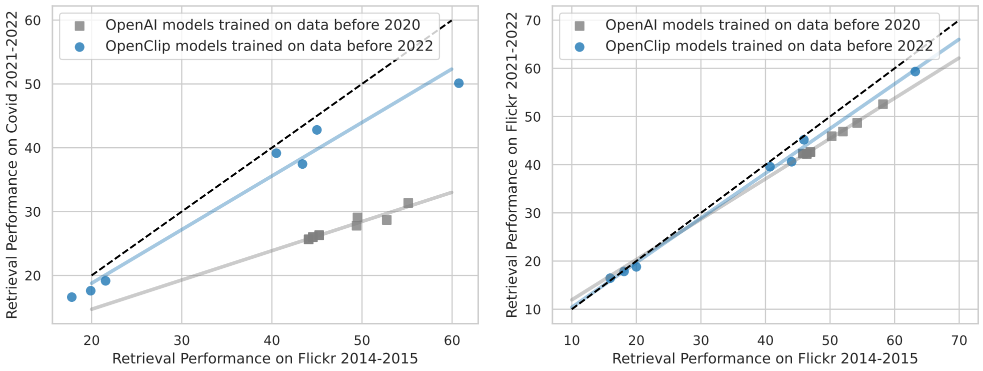


Figure J.12: (Left) Comparison of retrieval performance on COVID queries versus Flickr queries (construction described in App. J.3.5). (Right) Comparison on old Flickr versus new Flickr data. Clearly, we observe that while gap on old versus new flickr data is small, the gap is significantly larger on Covid queries.

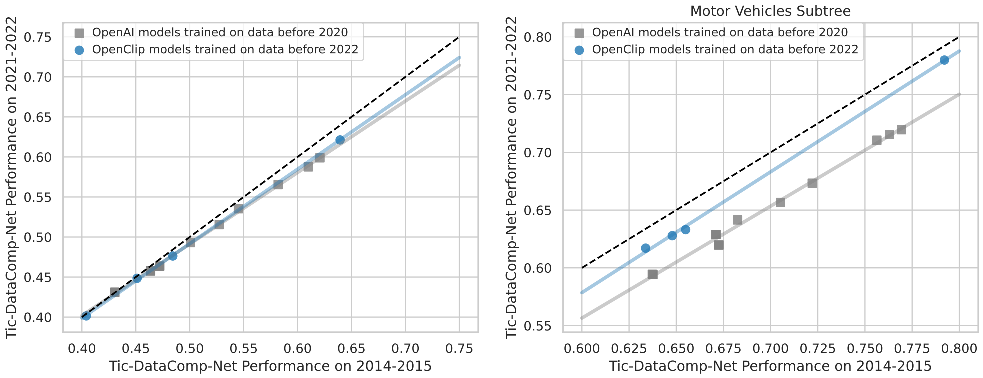


Figure J.13: (Left) Comparison on old versus new data from TIC-DataComp-Net. (Right) Comparison on motor vehicles node from TIC-DataComp-Net. For our classification task, we observe a very small drop (≈ 1%) when averaged across all categories. However, we observe a substantial gap on classes in “motor vehicle” subtree, when comparing OpenAI and OpenCLIP models. These findings highlight that while overall ImageNet classes may remain timeless, certain categories tend to evolve faster than others.

TIC-DataComp analysis through the lens of constructed evaluation tasks Here, we compare performance of OpenAI and OpenCLIP models on our datasets. We observe a significant performance gap between OpenAI and OpenCLIP models on our dynamic retrieval task (Fig. 11.1). This gap widens notably on retrieval queries where captions mention COVID-19. On the other hand, OpenAI and OpenCLIP models exhibit similar robustness for retrieval on data coming from Flickr highlighting that data from some domains do not exhibit shifts that cause performance drops. For our classification task, we observe a very small drop ($\approx 1\%$) when averaged across all categories. However, we observe a substantial gap on specific subtrees in ImageNet. For example, classes in “motor vehicle” subtree show an approximate 7% performance drop, when comparing OpenAI and OpenCLIP models. These findings highlight that while overall ImageNet classes may remain timeless, certain categories tend to evolve faster than others. Our qualitative and quantitative analysis on TIC-DataComp clearly highlights evolution of distributions and captures different properties than standard benchmarks.

Quantitative analysis on TIC-YFCC We analyze TIC-YFCC using off-the-shelf sentence and image encoders. For off-the-shelf sentence embedder, we used an existing sentence transformer from Hugging Face (Reimers and Gurevych, 2019). For the image encoder, we use a CLIP pretrained ViT-B-16 model (Ilharco et al., 2021; Radford et al., 2021).

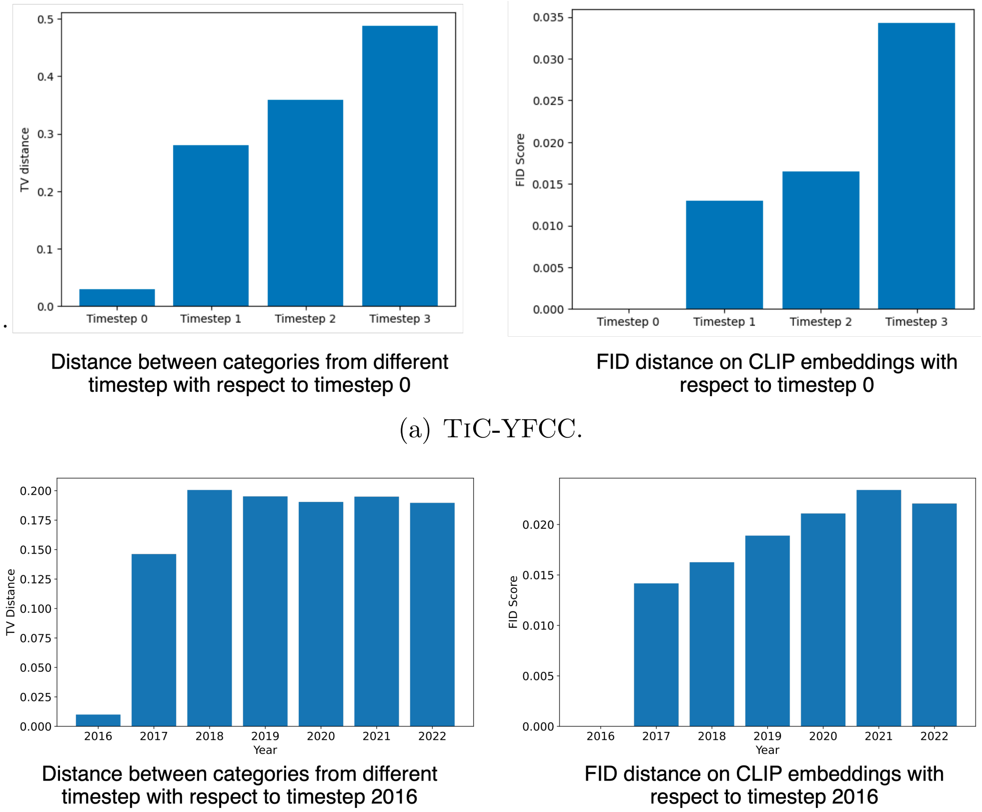
We first embed images from different time steps with an OpenAI CLIP encoder and then compute Frechet Inception Distance (FID; Seitzer (2020)). As time progresses, we observe that FID distance increases with respect to data from first time step (Fig. J.14). Similarly, we use the pretrained sentence transformer to extract top-5 categories from Wordnet Nouns for each caption. We then obtain a distribution over these Nouns for each time step. We observe that the TV distance over the distribution of WordNet nouns evolves over time when compared to data from the first time step.

J.3.7 Creation Pipiline for TIC-DataComp

We collect timestamps for the CommonPool dataset introduced in DataComp. We repeat the crawling process described in Gadre et al. (2023) to download WARC files from Common Crawl. In particular, we follow the same multistep process which involved: (i) parsing URLs and alt-text from Common Crawl dumps and downloading these images; (ii) tagging images with meta data and id of the common crawl batch; and (iii) conducting evaluation set duplication and safety content filtering. After downloading the WARC files, we perform a join with the datacomp 12.8B examples. During this join, we lost approximately 0.1B of examples that are no longer available online. Moreover, while performing this join, we only retain examples with their first occurrence. This is done before running any de-duplication on image-text pairs for exact matches as done in Gadre et al. (2023).

The source of DataComp is Common Crawl, which periodically releases web-crawled data snapshots, typically on a monthly basis since 2014 with new and updated webpages. This process provides timestamps at the granularity of months, spanning years 2014–2022.

We note that while this augmented time information may contain some noise, on average, we find it to be a reasonably accurate proxy for the upload time of web pages. To perform an initial check, we note that our data contains images from flickr which provides an API to query for true upload timestamp. So we extract 10k examples from our benchmark TIC-DataComp and query Flickr for their true timestamp. Fig. J.16 summarizes true timestamps with timestamps extracted from CC.



Distance between categories from different timestep with respect to timestep 0

FID distance on CLIP embeddings with respect to timestep 0

(a) TiC-YFCC.

Distance between categories from different timestep with respect to timestep 2016

FID distance on CLIP embeddings with respect to timestep 2016

(b) TiC-DataComp (M).

Figure J.14: Distribution shift results. Analysis on TiC-YFCC and TiC-DataComp (M) using off-the-shelf sentence and image encoders. We first embed images from different time steps with an OpenAI CLIP encoder and then compute Frechet Inception Distance (FID; Seitzer (2020)). As time progresses, we observe that FID distance increases with respect to data from first time step. Similarly TV distance over categorical distribution on Wordnet Noun synsets also increases with time when compared to categorical distribution on first timestep.

J.4 Additional Experimental Details

J.4.1 Additional details on ImageNet IID split continual learning experiment

With ImageNet data, we consider 2, 4 and 8 splits including the full dataset. This design is inspired by Ash and Adams (2020). We consider ViT-B/16 architecture trained for 300 epochs on full data and split the iterations corresponding to 300 epochs equally among k splits when training sequentially. We keep all other hyperparameters, such as learning rate, optimizer, and batch size, set to the standard values typically employed for training ViT-B/16 on the ImageNet dataset (Dosovitskiy et al., 2020). We also employ ℓ_2 regularization and augmentation on ImageNet training data. We evaluate the models on IID ImageNet test set.

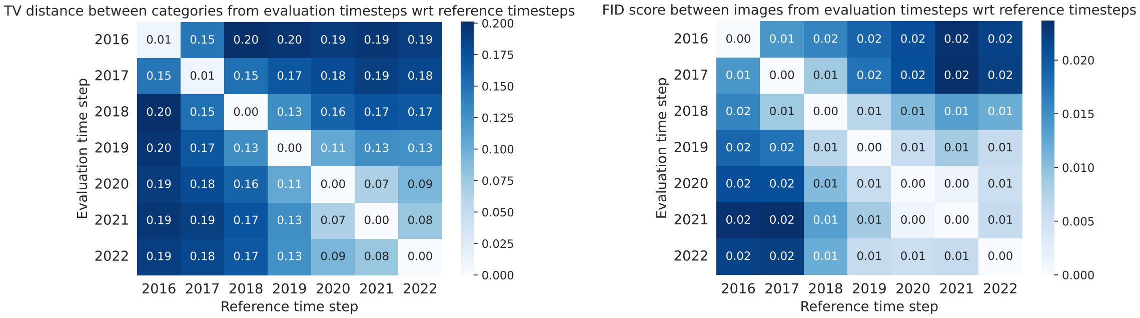


Figure J.15: Distribution shift analysis on TIC-DataComp (M) using off-the-shelf sentence and image encoders. We first embed images from different time steps with an OpenAI CLIP encoder and then compute Frechet Inception Distance (FID; Seitzer (2020)). As time progresses, we observe that FID distance increases with respect to data from first time step. Similarly TV distance over categorical distribution on Wordnet Noun synsets also increases with time when compared to categorical distribution on first timestep.

Our Imagenet experiments were primarily inspired by the "loss of plasticity" phenomenon described in Ash and Adams (2020). Their study demonstrates that models sequentially trained on two splits of CIFAR-10 data (initially on 50%, followed by 100% of data) exhibit poorer generalization compared to models trained from scratch on the entire dataset. Since we do not observe this behavior for continual training of CLIP, we investigated the existence of such behaviors on up to 8 splits of Imagenet. Our findings reveal that the simple cumulative baseline (with no extra budget) remains competitively close to the Oracle model (that benefits from using the full compute budget on the entire pooled training data from the beginning).

Prior works (Hu et al., 2021; Prabhu et al., 2023) performed continual learning experiments on Imagenet to compare different methods and highlight the effectiveness of continual training on synthetic continual learning setups derived from ImageNet. While these papers include results with an Oracle method, differences in the settings considered in these studies limit direct comparisons.

In particular, we show the performance gap of less than 1% in the same setup used otherwise in the paper when using SOTA training procedures achieving 81% validation performance. Comparitively the referenced Hu et al. (2021) does not show whether the 65% to 77% performance gap in their Table 1 can be bridged by increasing the compute for their method. Instead, authors show that if they restrict the compute for Oracle in Table 2, the Oracle performance drops to 68% (with ≈ 3% gap).

Moreover, in Prabhu et al. (2023), authors perform experiments on DI-Imagenet-2k where they start with an initial memory of Imagenet-1k 1.2 M samples and sequentially observe data for the same classes 1k classes from Imagenet-21k pool. This makes comparing streaming accuracy (or Imagenet-1k accuracy) for different methods incomparable with our setup (with a gap of over 7% in streaming accuracy even at step 8 as compared to less than 1% in our setup).

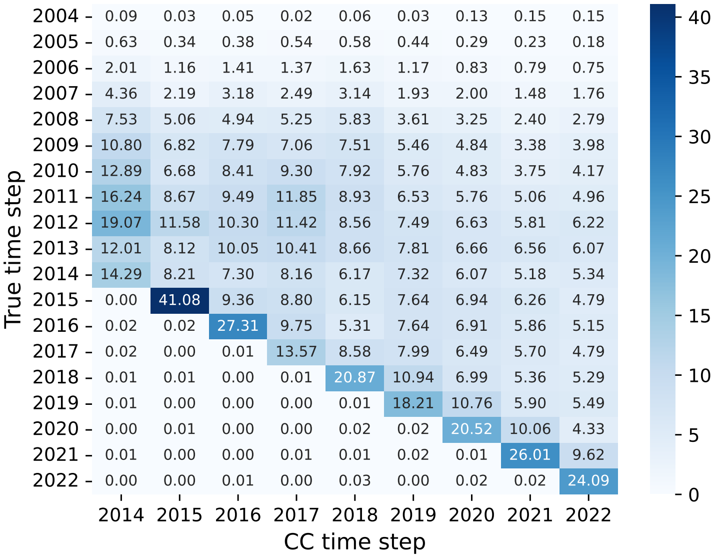


Figure J.16: Comparison of Common Crawl assigned timestamp and true timestamp on a subset of 10k examples containing image-text pairs from Flickr. We observe a clear trend where CC timestamps correlate with true timestamps.

J.4.2 Training and Hyperparameter Details

We create a common experimental setup by fixing the training procedure for sequential runs. Unless specified otherwise, we closely follow the CLIP training recipe proposed in (Ilharco et al., 2021; Radford et al., 2021) where we train models with a contrastive objective over images and captions. Given a set of image-text pairs, we train an image encoder and a text encoder such that the similarity between the representations of images and their corresponding text is maximized relative to unaligned pairs. Only LwF deviates from this standard training procedure. For each benchmark, we pick Vision Transformers (ViTs) as the image encoder, in particular, we fix the model architecture to ViT-B/16 (Dosovitskiy et al., 2021). We fix the Adam optimizer and its hyperparameters to values suggested in (Ilharco et al., 2021).

We primarily ablate over only two things: maximum learning rate with cosine learning schedule and warm up iterations for sequential training. For choosing other hyperparameters, we follow the OpenCLIP library (Ilharco et al., 2021).

J.4.3 Replay sizes with Exp and Equal strategies

We default to using 2D size of data where D represents incoming data size from new time step. As described in the main text, for -Exp, we reduce the buffer size by half of what we used at old time step and use rest of the half as data from previous time step. App. J.3.3 lists the dataset sizes for each benchmark which dictate the exact buffer sizes.

J.5 Results with Other Continual Learning Methods

J.5.1 Results with EWC Method

As proposed in the original work Kirkpatrick et al. (2017), we implement EWC method where we optimize the following loss:

$$\mathcal{L}_{EWC}(\theta) = \mathcal{L}(\theta) + \sum_i \frac{\lambda_{EWC}}{2} F_i (\theta_i - \theta_{t-1,i})^2 ,$$

where $\mathcal{L}(\theta)$ is the standard contrastive loss on data from time step t, F_i is the i-th diagonal entry of the fisher information matrix, and θ_{t-1} are the frozen parameters from previous time step. We perform experiments with different values of $\lambda_{EWC} \in \{1, 10, 100, 400\}$ (see Table J.15).

Table J.15: **Zero shot performance on our time-continual benchmarks with EWC.** * and ** denote methods that violate the compute budget. For static tasks, we tabulate accuracy of the models obtained on the final timestamp. For dynamic tasks, we tabulate forward/backward transfer and ID performance on retrieval tasks (Sec. 11.2.3). We observe that EWC performs worse than Sequential, Patching and LwF.

Benchmark	**Method**	**Compute** (MACs)	**Static Tasks**				**Dynamic Retrieval Tasks**		
			ImageNet	ImageNet dist. shift	Flickr30k	Average over 28 datasets	Backward Transfer	ID Performance	Forward Transfer
TIC-DataComp (M)	Sequential	3.0×10^{18}	19.2	16.4	16.4	15.0	25.7	26.4	14.9
	Patching	3.0×10^{18}	19.3	16.8	18.5	14.7	26.9	25.4	14.5
	LwF*	3.8×10^{18}	19.2	16.5	17.7	14.3	25.6	26.6	14.9
	EWC ($\lambda_{EWC} = 1$)*	3.6×10^{18}	18.7	16.3	16.2	15.1	25.5	26.4	14.8
	EWC ($\lambda_{EWC} = 10$)*	3.6×10^{18}	18.1	15.8	16.8	14.7	24.8	25.7	14.4
	EWC ($\lambda_{EWC} = 100$)*	3.6×10^{18}	17.6	15.4	16.3	14.8	24.4	25.4	14.3
	EWC ($\lambda_{EWC} = 400$)*	3.6×10^{18}	17.0	15.0	16.4	14.3	24.1	24.9	14.0

J.5.2 Results with Oversampling + Counting Based Sampling Method

In this section, we perform ablation on Cumulative-Equal. In particular, we made the following two modifications: (i) *Count based sampling*: Instead of random sampling, we implemented the count-based subsampling that prioritizes not/less used examples; (ii) *Oversampling*: We oversampled data from old timesteps with ratio inversely proportional to the ratio of examples, i.e., if the old data is of size D/2 and the new data is of size D, then we upsample old data with 2:1 ratio.

However, we observe that this method doesn't improve performance over Cumulative-Equal and in fact hurts the performance slightly (see Table J.16). We hypothesize that this can be due to a decreasing marginal utility of labeled data as highlighted in Cui et al. (2019). Their work argues that due to information overlap among data, as the number of samples increases, the marginal benefit a model can extract from the data diminishes. As a result, Cui et al. (2019) proposed using of "effective sample size" instead of the actual number of samples to obtain the ratio used to perform re-sampling or re-weighting. In particular, the expression of "effective sample size" is given by $E_n = \frac{1-\beta^n}{1-\beta}$ where n is the original sample size and β is a hyperparameter that Cui et al. (2019) selects from $\beta \in \{0.9, 0.99, 0.999, 0.9999\}$.

For different time steps, we leverage this expression of E_n to calculate the effective number of samples. In our settings (even at small scales), our datasets contain an order of 100k image-text pairs even after subsampling data from old time step. For example, with -Equal baseline, when training on the last time step (i.e., 2022), the smallest dataset (i.e., 2016) is of approximately 400k samples. Plugging in the expression for effective sample size from Cui et al. (2019), we observe that for all $\beta \in (0, 0.99999)$, the ratio of effective sample sizes for different time steps remains close to 1. This may highlight why our naive over-sampling strategy doesn't improve over no-oversampling.

Table J.16: **Zero shot performance on our time-continual benchmarks with oversampling and counting-based sampling.** * and ** denote methods that violate the compute budget. For static tasks, we tabulate accuracy of the models obtained on the final timestamp. For dynamic tasks, we tabulate forward/backward transfer and ID performance on retrieval tasks (Sec. 11.2.3).

Benchmark	**Method**	**Compute** (MACs)	Static Tasks				Dynamic Retrieval Tasks		
			ImageNet	ImageNet dist. shift	Flickr30k	Average over 28 datasets	Backward Transfer	ID Performance	Forward Transfer
TIC-DataComp (M)	Sequential	3.0×10^{18}	19.2	16.4	16.4	15.0	25.7	26.4	14.9
	Cumulative-Equal (Counts + OS)	3.0×10^{18}	18.1	15.3	14.3	16.5	28.9	23.7	14.2
	Cumulative-Equal	3.0×10^{18}	22.1	18.4	19.2	17.1	31.8	26.8	15.1

J.6 Results With New Evaluation Metrics on Dynamic Tasks

Recall, T represent the number of time steps for which we have data. For each training method, we generate a total of T models, each corresponding to the end of training at a particular time step. For each model and a dynamic evaluation task, we obtain T performance values. We represent these values using the performance matrix $\mathcal{E}$, where each entry $\mathcal{E}_{i,j}$ signifies the performance of the model obtained after observing training data at time step i when evaluated on a dataset from time step j. Defining backward metrics as in Sec. 11.2.2 involves averaging the entries in the upper and lower diagonal of our performance matrix $\mathcal{E}$, i.e., it was calculated as the average of time steps before each training step (i.e., the lower triangular of $\mathcal{E}$), i.e., $\frac{\sum_{i \geqslant j} \mathcal{E}_{ij}}{(T(T-1))/2}$. This backward transfer metric has been used in prior works Lin et al. (2021). However, this approach inadvertently resulted in the backward transfer metric being influenced by later evaluation time steps resulting in backward transfer performance numbers slightly larger than ID performance.

To address this issue, we've revised our metric calculation method to metric as in Díaz-Rodríguez et al. (2018). Now, we normalize the data in each row, which corresponds to evaluation time steps by subtracting the ID performance. This adjustment ensures a more balanced and accurate representation across all training time steps. In particular, our updated forward and backward transfer metrics can be summarized as:

Table J.17: **Zero shot performance on our time-continual benchmarks.** * and ** denote methods that violate the compute budget. For dynamic tasks, we tabulate forward/backward transfer and ID performance on retrieval tasks with updated metrics as defined in App. J.6.

Benchmark	**Method**	**Compute** (MACs)	**Dynamic Retrieval Tasks**				
			Backward Transfer	ID Performance	Forward Transfer	Relative Backward Transfer	Relative Forward Transfer
TiC-YFCC	Restart	3.4×10^{18}	13.2	41.4	18.6	−29.8	−21.2
	Sequential	3.4×10^{18}	42.2	48.4	23.7	−9.5	−21.5
	Patching	3.4×10^{18}	44.7	53.4	24.5	−15.6	−22.0
	Cumulative-Exp	3.4×10^{18}	60.4	60.1	27.1	−9.8	−23.0
	Cumulative-Equal	3.4×10^{18}	60.4	60.4	27.1	−10.3	−23.0
	Cumulative-All	3.4×10^{18}	**66.4**	**60.2**	**27.6**	−4.1	−22.4
	LwF*	4.1×10^{18}	36.6	56.0	23.2	−27.4	−24.9
	Cumulative-All*	3.6×10^{18}	**66.8**	**60.3**	**27.6**	−3.9	−22.4
	Oracle**	8.5×10^{18}	**66.1**	**61.8**	**26.9**	−6.6	−24.0
TiC-RedCaps	Restart	3.4×10^{18}	21.3	25.4	22.4	−4.5	−2.7
	Sequential	3.4×10^{18}	33.0	33.6	27.5	−3.8	−3.0
	Patching	3.4×10^{18}	34.8	34.8	27.8	−3.9	−3.0
	Cumulative-Exp	3.4×10^{18}	44.5	42.0	32.6	−3.0	−4.0
	Cumulative-Equal	3.4×10^{18}	44.4	42.0	32.6	−3.0	−4.0
	Cumulative-All	3.4×10^{18}	**48.9**	**43.2**	**33.4**	−0.6	−3.5
	LwF*	4.1×10^{18}	35.4	36.0	28.4	−4.6	−3.7
	Cumulative-All*	3.6×10^{18}	**49.0**	**43.4**	**33.4**	−1.0	−3.5
	Oracle**	8.5×10^{18}	**48.5**	**43.1**	**33.4**	−1.0	−3.4
TiC-DataComp (M)	Sequential	3.0×10^{18}	25.7	26.4	14.9	−4.7	−7.6
	Patching	3.0×10^{18}	26.9	25.4	14.5	−1.9	−7.4
	Cumulative-Exp	3.0×10^{18}	31.7	27.1	**15.2**	0.3	−7.6
	Cumulative-Equal	3.0×10^{18}	31.8	26.8	15.1	0.9	−7.6
	Cumulative-All	3.0×10^{18}	33.8	26.4	15.1	3.5	−7.3
	LwF*	3.8×10^{18}	25.6	26.6	14.9	−4.8	−8.0
	Cumulative-All*	3.9×10^{18}	**36.7**	**28.3**	**15.5**	3.0	−7.3
	Oracle**	1.2×10^{19}	34.9	27.8	**15.6**	2.5	−7.7
TiC-DataComp (L)	Sequential	2.7×10^{19}	52.6	**58.4**	41.1	−8.7	−14.4
	Patching	2.7×10^{19}	55.2	57.5	40.9	−4.9	−13.9
	Cumulative-Exp	2.7×10^{19}	60.4	**58.4**	**41.4**	−1.1	−13.8
	Cumulative-Equal	2.7×10^{19}	60.9	**58.2**	**41.4**	−0.3	−13.8
	Cumulative-All	2.7×10^{19}	62.1	57.3	41.2	2.2	−13.5
	Cumulative-All*	4.1×10^{19}	63.0	57.8	41.2	2.1	−13.5
	Oracle**	1.1×10^{20}	**64.3**	**58.6**	**41.8**	2.2	−13.3
TiC-DataComp (XL)	Sequential	2.7×10^{20}	63.1	68.9	56.8	−5.6	−12.3
	Cumulative-All	2.7×10^{20}	**70.7**	**68.5**	**57.1**	2.5	−11.7
	Cumulative-All*	3.5×10^{20}	**71.0**	**68.6**	**57.1**	2.5	−11.7

- *Backward transfer*: Let $\mathcal{B}_i$ denote the average performance on evaluation tasks before time i, then we define backward transfer as average of $\mathcal{B}_i$ across each training step, i.e., $\sum_{i=2}^{T} \frac{\sum_{i \geqslant j} \mathcal{E}_{ij} - \mathcal{E}_{ii}}{T(T-1)/2}$

- *Forward transfer*: Let $\mathcal{F}_i$ denote the average performance on evaluation tasks after time i, then we define forward transfer as average of $\mathcal{F}_i$ across each training step, i.e., $\sum_{i=1}^{T-1} \frac{\sum_{i \leqslant j} \mathcal{E}_{ij} - \mathcal{E}_{ii}}{T(T-1)/2}$

Appendix K

Appendix: Prompting is a Double-Edged Sword: Improving Worst-Group Robustness of Foundation Models

K.1 Extended Related Works

Theoretically analyzing robustness of self-supervised learning. While several works theortically analyze (HaoChen and Ma, 2022; HaoChen et al., 2021; Mitrovic et al., 2020; Saunshi et al., 2022; Tian et al., 2020; Wang and Isola, 2020) models pretrained with contrastive learning, masked image and language modeling, they mainly do this for few-shot in-distribution generalization on downstream tasks. In contrast, there are fewer works that focus on out-of-distribution robustness (HaoChen et al., 2022; Kumar et al., 2022a; Shen et al., 2022), and even fewer on robustness to spurious correlations (Garg et al., 2023b), and all of them do this for unimodal few-shot settings. In contrast, we theoretically analyse zero-shot generalization for multimodal contrastive learning. (Chen et al., 2023; Zhang et al., 2023) are recent works that also theoretically analyze the multimodal setting, and the former only studies few-shot in-distribution generalization, similar to Lee et al. (2021). Closest to our analysis is Zhang et al. (2023), which analyzes zero-shot performance of CLIP, but unlike us they do not specifically model the pretraining distribution to also include spurious attributes from the downstream task, which we show impacts robustness to spurious correlations.

K.2 Proofs for our theoretical results

K.2.1 Worst group guarantees for PfR

Theorem K.2.1 (PfR's worst group error; restated)**.** *For PfR output $\widehat{f}$, w.h.p. $1-\delta$, worst group generalization error of $\widehat{f}$ is $\lesssim \sqrt{\log \mathfrak{C}(\mathcal{F})K/\delta / n} + \mathrm{err}_c(\mathrm{FM}(t_c))$, where $\mathfrak{C}(\mathcal{F})$ is* *complexity of $\mathcal{F}$, K is number of groups and latter term is* FM*'s zero-shot performance on confounder prediction.*

Proof. Recall the objective for PfR which minimizes worst group loss over predicted groups $\widehat{G}_1, \ldots, \widehat{G}_K$. Let,

$$f^\star \;:=\; \inf_{f \in \mathcal{F}} \sup_{k \in [K]} \mathbb{E}_{P_T}\left[l(h(\mathbf{x}), \mathrm{y}) \mid (\mathbf{x}, \mathrm{y}) \in \widehat{G}_k\right] \tag{K.1}$$

Lemma K.2.2 (worst-case risk generalization (Group DRO))**.** *With probability $\geqslant 1 - \delta$ over dataset $\mathcal{D} \sim P^n$, the worst group risk for $f^\star$ can be upper bounded by the following, where* opt *is the minimum on the training objective,*

$$\sup_{k \in [K]} \mathbb{E}_{P_T}\left[l(h(\mathbf{x}), \mathrm{y}) \mid (\mathbf{x}, \mathrm{y}) \in \widehat{G}_k\right] \;\lesssim\; \mathrm{opt} + \sqrt{\frac{\log\left(\frac{\mathfrak{C}K}{\delta}\right)}{n}}\;,$$

*where $\mathfrak{C}$ is the complexity of class $\mathcal{F}$ (*e.g. *the covering number ()).*

Proof. We first apply the generalization bound for a single group, which is given by $\sqrt{\frac{\log\left(\frac{\mathfrak{C}}{\delta}\right)}{n}}$ (Wainwright, 2019), followed by a union bound over the K groups. □

We can break down down the worst group loss for the learned function $\widehat{f}$ on the true groups $G_1, \ldots, G_K$ in the following way, where we assume loss ℓ is M bounded:

$$\sup_{k \in [K]} \mathbb{E}_{P_T}\left[l(\widehat{f}(\mathbf{x}), \mathrm{y}) \mid (\mathbf{x}, \mathrm{y}) \in G_k\right] \leqslant \sup_{k \in [K]} \mathbb{E}_{P_T}\left[l(\widehat{f}(\mathbf{x}), \mathrm{y}) \mid (\mathbf{x}, \mathrm{y}) \in G_k \cap \widehat{G}_k\right] \tag{K.2}$$

$$+ M\mathbb{E}_{P_T}\left[\mathbb{1}(x \in \widehat{G}_k) \mid x \in G_k\right] \tag{K.3}$$

$$+ M\mathbb{E}_{P_T}\left[\mathbb{1}(x \in G_k) \mid x \in \widehat{G}_k\right] \tag{K.4}$$

Since $\max_{1,2}(a_1 + b_1, a_2 + b_2) \leqslant \max_{1,2}(a_1, a_2) + \leqslant \max_{1,2}(b_1, b_2)$ for some scalars a_1, a_2, b_1, b_2, we can upper bound $\sup_{k \in [K]} \mathbb{E}_{P_T}\left[l(\widehat{f}(\mathbf{x}), \mathrm{y}) \mid (\mathbf{x}, \mathrm{y}) \in G_k\right]$ as:

$$\begin{aligned}\sup_{k \in [K]} \mathbb{E}_{P_T}\left[l(\widehat{f}(\mathbf{x}), \mathrm{y}) \mid (\mathbf{x}, \mathrm{y}) \in G_k\right] &\leqslant \sup_{k \in [K]} \mathbb{E}_{P_T}\left[\mathbb{1}(x \in \widehat{G}_k) \mid x \in \widehat{G_k}\right] + \mathbb{E}\left[\mathbb{1}(\mathrm{FM}(x, t_c) \neq c)\right] \\ &= \sup_{k \in [K]} \mathbb{E}_{P_T}\left[\mathbb{1}(x \in \widehat{G}_k) \mid x \in \widehat{G_k}\right] + \mathrm{err}_{\mathrm{sp}}^{\mathrm{avg}}(\mathrm{FM}(x, t_c)).\end{aligned}$$

for positive losses. Above, we replaced the group mixmatch error with the error of the zero-shot classifier $\mathrm{FM}(x, t_c)$. Further, in our case $M = 1$.

The above result when used in a simple triangle inequality with the result in Lemma K.2.2 completes the proof of Theorem K.2.2.

□

K.2.2 Analysis of multimodal contrastive pretraining

Before, we present our the proofs for our main theoretical result, we will prove a key Lemma that allows us to derive general solutions for multimodal spectral contrastive loss in Equation (12.2), done on any class of ϕ, ω.

General solution for any function class

Lemma K.2.3 (General solutions for multimodal contrastive learning)**.** *When ϕ, ω are restricted to orthonormal functions in $L^2(P)$, then the objective in Equation (12.2) is equivalent to $\min_{\phi,\omega} \int_x \phi(x)\sqrt{q(x)}A(w(t)\sqrt{q(t)})(x)\,\mathrm{d}x$. Here, $A(f(t))$ is the linear operator*

$$A(f(t)) =: \int_t {}^{p(x,t)f(t)}\!/\!_{\sqrt{q(x)q(t)}}\,\mathrm{d}t,$$

and A^+ is its adjoint. Its adjoint is then:

$$A^+(g(x)) =: \int_x {}^{p(x,t)g(x)}\!/\!_{\sqrt{p(x)p(t)}}\,\mathrm{d}t.$$

Given the constraints on ϕ, ω, to be orthonormal and operators A, A^+ in Proposition K.2.3, the optimal solutions for (12.2) are $\phi_i(x) = {}^{f_i(x)}\!/\!_{\sqrt{p(x)}}$ and $\omega_i(t) = {}^{g_i(t)}\!/\!_{\sqrt{p(t)}}$, where $\{f_i\}_{i=1}^k$ and $\{g_i\}_{i=1}^k$ are the top k eigen functions of self-adjoint AA^+ and A^+A respectively.

Proof. First, we break down the spectral contrastive loss in the following way where q is the density of the measure $Q(x,t)$:

$$\begin{aligned} & -2\mathbb{E}\left[\phi(x)^\top\omega(t)\right] + \mathbb{E}_x\mathbb{E}_t(\phi(x)^\top\omega(t))^2 && \text{(K.5)} \\ & = \int_{\mathcal{X},\mathcal{T}} \left(\frac{Q(x,t)}{\sqrt{q(x)}\sqrt{q(t)}} - \sqrt{Q(x)}\phi(x)^\top\omega(t)\sqrt{q(t)}\right)^2 dxdt + \text{const.} && \text{(K.6)} \end{aligned}$$

Then consider the case where the output dimension is 1. We consider the constrained objective where $\int_{\mathcal{X}} \phi^2(x)\,dx = 1$ and $\int_{\mathcal{T}} \omega^2(t)\,dt = 1$. Plugging this in, we conclude the above objective is equivalent to: to $A(\widetilde{\omega})(x) = \int \frac{q(x)q(t)}{\sqrt{q(x)q(t)}}\widetilde{\omega}(t)dt$. Here:

$$\widetilde{\omega}(t) = \omega(t)\sqrt{q(t)} \quad \widetilde{\phi}(x) = \sqrt{q(x)}\phi(x) \tag{K.7}$$

Following Eckart and Young (1936), we know that the solution to the above optimization problem is given by the eigenvectors of the self-adjoint operators $AA^\dagger$ and $A^\dagger A$.

□

For the multimodal spectral contrastive loss in Equation (12.2), when we additionally require the image and text encoders to be normalized in $L_2(P)$, (*i.e.* any $f : \mathcal{X} \mapsto \mathbb{R}$ or $f : \mathcal{T} \mapsto \mathbb{R}$ such that $\int f^2 \mathrm{d}P < \infty$), then the objective can be redefined with the linear operator A in Lemma K.2.3.

Leveraging the result above, we closely analyze the impact the of the distribution skew by deriving closed form solutions for ϕ, ω when they are restricted to the class of linear functions. Note, given the one hot encoding of the text in $\mathcal{T}$ the linearity assumption in no way restricts the class of text encoders. We present our result in Theorem K.2.4.

Proof of Theorem 12.4.2

Theorem K.2.4 (Optimal solution for spectral contrastive loss)**.** *Let $p \geqslant p_0 > 0.5$ for some fixed p_0 and $\phi = \mathbf{A}^\top x$, $\omega = \mathbf{B}^\top t$ are linear with $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{k \times d}$. Then, under slightly stricter constraints on ϕ, the solutions $\mathbf{A}^\star, \mathbf{B}^\star$ for the objective in (12.2), are the top k columns of the matrix on the left and right respectively, where* $\tan(2\theta) = \frac{4\gamma\alpha(\gamma^2/\nabla^2+1)}{((2p-1)+1/2p-1)}$ *and $\mathbf{U}_{d_\text{n}} \in \mathbb{R}^{d_\text{n} \times d_\text{n}}$ is unitary.*

$$\begin{bmatrix} \cos(\theta)/\sqrt{\nabla^2+\gamma^2} & \sin(\theta)/\sqrt{\nabla^2+\gamma^2} & \mathbf{0}_{d_\text{n}}^\top \\ -\sin(\theta)/\alpha & \cos(\theta)/\alpha & \mathbf{0}_{d_\text{n}}^\top \\ \mathbf{0}_{d_\text{n}} & \mathbf{0}_{d_\text{n}} & \mathbf{U}_{d_\text{n}} \end{bmatrix}, 0.5 \begin{bmatrix} +1 & +1 & +1 & -1 \\ +1 & +1 & -1 & +1 \\ +1 & -1 & -1 & -1 \\ +1 & -1 & +1 & +1 \end{bmatrix}.$$

In the above statement, $\alpha = \gamma = 1$.

Proof. Recall from Lemma K.2.3, the general solutions are given by eigen functions of $AA^\dagger$, and $A^\dagger A$. For linear functions, that are norm regularized, *i.e.* $\mathbb{E}[\phi(x)\phi(x)^\top] = I_k$ and $\mathbb{E}[\omega(t)\omega(t)^\top] = I_k$, we derive the following objective:

$$\max_{\phi : \phi^\top \Sigma \phi = 1} \phi^\top \widetilde{\Sigma} \phi,$$

$$\Sigma = \mathbb{E}[xx^\top] \quad \widetilde{\Sigma} = \mathbb{E}_t[\mathbb{E}[x|t]\mathbb{E}[x|t]^\top].$$

Here, we encode text as a one-hot vector: Thus, the set of text descriptions $\mathcal{T}$ is: { "y is $+1$", "c is $+1$", "c is -1" and "y is -1" }, which we input as one hot encodings $[1,0,0,0]^\top, [0,1,0,0]^\top, [0,0,1,0^\top]$ and $[0,0,0,1]^\top$ respectively to the text encoder ω.

$$\max_{\phi : \omega^\top \Sigma_t \omega = 1} \omega^\top \widetilde{\Sigma_t} \omega,$$

$$\Sigma_t = \mathbb{E}[tt^\top] \qquad \widetilde{\Sigma}_t = \mathbb{E}_x[\mathbb{E}[t|x]\mathbb{E}[t|x]^\top].$$

Since both are identical but involve different matrices, we show our working for one, and plug in values from the distribution for the other.

First we note that changing the constraint to $\phi^\top \Sigma \phi \leqslant 1$, does not change the optimal solution, since these are eigen vectors and Σ is full rank in both cases. Second, we recall the identity:

$$\phi^\top \Sigma \phi \leqslant 2 \cdot \phi^\top \text{diag} \Sigma \phi.$$

Thus, we replace the constraint on ϕ, with the right right hand side of the above expression. Note that, whenever the right hand side $\leqslant 1/2$, our original constrained is satisfied. So, we solve this more regularized objective for conveniece of obtaining a more precise closed form solution.

Recall that in our setup both $\widetilde{\Sigma}$ and Σ are positive definite and invertible matrices. To solve the above problem, let's consider a re-parameterization: $\phi' = \text{diag}(\Sigma)^{1/2}\phi$, thus $\phi^\top \text{diag}(\Sigma)\phi = 1$, is equivalent to the constraint $\|\phi'\|_2^2 = 1$. Based on this re-parameterization we are now solving:

$$\underset{\|\phi'\|_2^2=1}{\arg\max} \quad \phi'^\top \text{diag}(\Sigma)^{\frac{-1}{2}} \cdot \widetilde{\Sigma} \cdot \text{diag}(\Sigma)^{-1/2}\phi', \tag{K.8}$$

which is nothing but the top eigenvector for $\text{diag}(\Sigma)^{-1/2} \cdot \widetilde{\Sigma} \cdot \text{diag}(\Sigma)^{-1/2}$.

Now, to extend the above argument from $k = 1$ to $k > 1$, we need to care of one additional form of constraint in the form of feature diversity: $\phi_i^\top \Sigma_A \phi_j = 0$ when $i \neq j$. But, we can easily redo the reformulations above and arrive at the following optimization problem:

$$\underset{\substack{\|\phi_i'\|_2^2 = 1, \ \ \forall i \\ \phi_i'^\top \phi_j' = 0, \ \ \forall i \neq j}}{\arg\max} \qquad \left[\phi_1', \phi_2', \ldots, \phi_k'\right]^\top \text{diag}(\Sigma)^{-1/2} \cdot \widetilde{\Sigma} \cdot \text{diag}(\Sigma)^{-1/2} \left[\phi_1', \phi_2', \ldots, \phi_k'\right], \tag{K.9}$$

where $\phi_i' = \text{diag}(\Sigma)^{1/2}\phi_i$. The above is nothing but the top k eigenvectors for the matrix $\text{diag}(\Sigma)^{-1/2} \cdot \widetilde{\Sigma} \cdot \text{diag}(\Sigma)^{-1/2}$.

Let SVD_k is the top k singular vectors of an SVD decomposition. Now, from our problem description we state values of the four matrices above. For the image encoder, the solution is given by:

$$(\Sigma)^{-1/2}\text{SVD}_k(\text{diag}(\Sigma)^{-1/2} \cdot \widetilde{\Sigma} \cdot \text{diag}(\Sigma)^{-1/2})$$

where $\Sigma, \widetilde{\Sigma}$ are defined as follows:

$$\Sigma =: \begin{bmatrix} 1 + \nabla^2 & 2p - 1 & \mathbf{0}_{d_n} \\ 2p - 1 & 1 & \mathbf{0}_{d_n} \\ \mathbf{0}_{d_n}^\top & \mathbf{0}_{d_n}^\top & I_k \end{bmatrix} \tag{K.10}$$

$$\widetilde{\Sigma} =: \begin{bmatrix} (1 + (2p-1)^2)/2 & 2p - 1 & \mathbf{0}_{d_n} \\ 2p - 1 & (1 + (2p-1)^2)/2 & \mathbf{0}_{d_n} \\ \mathbf{0}_{d_n}^\top & \mathbf{0}_{d_n}^\top & I_k \end{bmatrix}.$$

On the other hand, for the text encoder, it is given by:

$$(\Sigma_t)^{-1/2}\text{SVD}_k(\text{diag}(\Sigma_t)^{-1/2} \cdot \widetilde{\Sigma_t} \cdot \text{diag}(\Sigma_t)^{-1/2})$$

$\Sigma_t = I_4$ and $\widetilde{\Sigma}$ is:

$$\widetilde{\Sigma} =: \begin{bmatrix} 1 & p & 1-p & 0 \\ p & 1 & 0 & 1-p \\ 1-p & 0 & 1 & p \\ 0 & 1-p & p & 1 \end{bmatrix}$$

Lemma K.2.5 (closed-form expressions for eigenvalues and eigenvectors of $\Sigma, \widetilde{\Sigma}$)**.** *For a 2×2 real symmetric matrix $\begin{bmatrix} a, & b \\ c, & d \end{bmatrix}$ the eigenvalues λ_1, λ_2 are given by the following expressions:*

$$\lambda_1 = \frac{(a+b+\delta)}{2}, \quad \lambda_2 = \frac{(a+b-\delta)}{2},$$

where $\delta = \sqrt{4c^2 + (a-b)^2}$. Further, the eigenvectors are given by $U = \begin{bmatrix} \cos(\theta), -\sin(\theta) \\ \sin(\theta), \ \cos(\theta) \end{bmatrix}$, where:

$$\tan(\theta) = \frac{b-a+\delta}{2c}.$$

For full proof of these statements see (Deledalle et al., 2017).

Plugging the above expressions into Lemma K.2.5 gives us the final solution and completes the proof.

□

Proof of Theorem 12.4.1

Theorem K.2.6. *(zero-shot robustness; restated) Let the zero-shot label (f) and confounder classifier (g) be obtained by minimizing the loss in (12.2) on infinite pretraining data. Then, for $\nabla = \Omega(1)$, label classifier is worse than random on the worst group, since $\mathrm{err}_{\mathrm{y}}^{\mathrm{wg}}(f) = {}^1\!/\!_2\,\mathrm{erfc}(-c_1 \nabla p)$. On the other hand, the confounder classifier suffers small error on all groups since $\mathrm{err}_{\mathrm{sp}}^{\mathrm{wg}}(g) = {}^1\!/\!_2\,\mathrm{erfc}(c_2 \nabla p)$. Here, $c_1, c_2 > 0$ are constants .*

Proof. First, we state the formal version of the theorem statement. Let f be zero-shot label predictor, and g be the zero-shot confounder predictor extracted from ϕ, ω in Theorem K.2.4. Then, the worst group error for f is:

$$\mathrm{err}_{\mathrm{y}}^{\mathrm{wg}}(f) = {}^1\!/\!_2 \cdot \mathrm{erfc}\left({}^{\rho}\!/\!_{\sqrt{2}}\right),$$

and for g is:

$$\mathrm{err}_{\mathrm{sp}}^{\mathrm{wg}}(g) = {}^1\!/\!_2 \cdot \mathrm{erf}\left({}^{\rho}\!/\!_{\sqrt{2}}\right),$$

where $\rho = -{}^1\!/\!_\nabla - \cot(\theta)\sqrt{{}^1\!/\!_{\nabla^2} + 1}$. Here, θ is the value defined in Theorem K.2.4.

Using our expressions for the zero-shot predictor in Sec. 12.4, we use the result from Theorem K.2.4 to define:

$$f([x_{\mathrm{r}}, x_{\mathrm{c}}]) = g([x_{\mathrm{r}}, x_{\mathrm{c}}]) = 2\mathbb{1}(-\frac{2x_{\mathrm{r}}\sin\theta}{\sqrt{1+\sigma^2}} + 2x_{\mathrm{c}}\cos\theta) - 1$$

Now, based on the signs along x_{r} and x_{c}, we conclude that the worst group for f is $y = 1, c = -1$.

$$\begin{aligned}
&\Pr(f([x_{\mathrm{r}}, x_{\mathrm{c}}]) \leqslant 1 \mid (y, c) = (1, -1)) \\
&= \Pr(\frac{-2\sin\theta}{\sqrt{1+\nabla^2}} \leqslant -2\cos\theta) \\
&= \Pr(\frac{x_{\mathrm{r}} - 1}{\nabla} \geqslant -\frac{1}{\nabla} + \sqrt{\frac{1}{\nabla^2} + 1}\cot\theta\cos\theta) \\
&= \frac{1}{2}\mathrm{erfc}(\frac{-1}{\nabla} - \cot\theta(\sqrt{1 + 1/\nabla^2})) \\
&= \frac{1}{2}\mathrm{erfc}(\rho/\sqrt{2}).
\end{aligned}$$

On the other hand the worst group for the confounder is $(y, c) = (1, 1)$, but even here, the error is negligible.

$$\begin{aligned}
&\Pr(f([x_{\mathrm{r}}, x_{\mathrm{c}}]) \leqslant 1 \mid (y, c) = (1, 1)) \\
&= \frac{1}{2}\mathrm{erfc}(\frac{-1}{\nabla} + \cot\theta(\sqrt{1 + 1/\nabla^2})) \\
&= \frac{1}{2}\mathrm{erf}(\rho/\sqrt{2}).
\end{aligned}$$

This completes our proof.

□

K.3 Experimental Setup

K.3.1 Prompts used for Zero-Shot Label Inference

CivilComments:

Predict label, ignore confounder:

Toxicity Prompt:

You are an AI bot that is an expert at detecting toxic comments. You are not biased by references to religion, race, and gender. Evaluate the following USER COMMENT on a news website, and classify the comment as either 'TOXIC' or 'SAFE', ignoring references to religion, race, and gender:

USER COMMENT: {}

In one word, the above comment is TOXIC (ONLY as ’YES’ or ’NO’):

Predict label:

Toxicity Prompt:

You are an AI bot that is an expert at detecting toxic comments. You are not biased by references to religion, race, and gender. Evaluate the following USER COMMENT on a news website, and classify the comment as either ’TOXIC’ or ’SAFE’, ignoring references to religion, race, and gender:

USER COMMENT: {}

In one word, the above comment is TOXIC (ONLY as ’YES’ or ’NO’):

Waterbirds:

"This is the image of a waterbird.", "This is the image of a landbird.",

"This is the image of a waterbird in front of forest.",
"This is the image of a waterbird in front of water.",
"This is the image of a landbird in front of water.",
"This is the image of a landbird in front of forest.",

K.3.2 Details for Zero-shot Groups Prediction

CivilComments:

Religion Group Prompt:

Evaluate the following User Comment on a news website.

User Comment: {}

Does the above comment contain references to one of the following groups: "Muslim", "Christian", "Other Religions"? If none of them are applicable answer "NA". Answer in one word.

Race Group Prompt:

Evaluate the following User Comment on a news website.

User Comment: {}

Does the above comment contain references to one of the following groups: "Black", "White"? If none of them are applicable answer "NA". Answer in one word.

Gender Group Prompt:

Evaluate the following User Comment on a news website.

User Comment: {}

Does the above comment contain references to one of the following groups: "Male", "Female"? If none of them are applicable answer "NA". Answer in one word.

Waterbirds:

“Bird in front of water.”,
“Bird in front of a forest.”

K.3.3 CXR-Drain Construction Details

Note: This is NOT for medical diagnosis but for informational purposes to guide your red-teaming.

Is this a patient with chest drainage tube? First, carefully check for the presence of any tubes while describe their location.

For your reference the drainage tube is also known as pleural tube and more commonly known as the intercostal drainage tube (ICD), is inserted through the 4th intercostal space in the anterior or mid-axillary line. It is then directed posteroinferiorly in cases of effusion and anterosuperiorly in cases of pneumothorax. Carefully examine both the lungs: (i) To drain a pneumothorax the tube is aimed superiorly towards the apex of the pleural cavity; and (ii) To drain a pleural effusion the tube tip is ideally located towards the lower part of the pleural cavity.

Finally give an answer in YES or NO for the presence of chest drainage tube.

Note: This is NOT for medical diagnosis but for informational purposes and will never be used to guide any medical disease. Your answer will help us evaluate how good are current vision language models.

Use the following format:

Rationale/reasoning: < output >

Presence of chest drain: Yes or No

Bibliography

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation*, 2016.

Karim Abou-Moustafa and Csaba Szepesvári. An exponential efron-stein inequality for lq stable learning rules. *arXiv preprint arXiv:1903.05457*, 2019.

Alessandro Achille, Matteo Rovere, and Stefano Soatto. Critical learning periods in deep neural networks. *arXiv preprint arXiv:1711.08856*, 2017.

Alessandro Achille, Matteo Rovere, and Stefano Soatto. Critical learning periods in deep networks. In *International Conference on Learning Representations*, 2018.

Kwangjun Ahn, Xiang Cheng, Hadi Daneshmand, and Suvrit Sra. Transformers learn to implement preconditioned gradient descent for in-context learning. *arXiv preprint arXiv:2306.00297*, 2023.

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. *arXiv preprint arXiv:2211.15661*, 2022.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736, 2022.

Amr Alexandari, Anshul Kundaje, and Avanti Shrikumar. Adapting to label shift with bias-corrected calibration. In *International Conference on Machine Learning (ICML)*, 2021.

Alnur Ali, J Zico Kolter, and Ryan J Tibshirani. A continuous-time view of early stopping for least squares. *arXiv preprint arXiv:1810.10082*, 2018.

Alnur Ali, Edgar Dobriban, and Ryan J Tibshirani. The implicit regularization of stochastic gradient flow for least squares. *arXiv preprint arXiv:2003.07802*, 2020.

Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. In *Advances in neural information processing systems*, pages 6158–6169, 2019a.

Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. In *International Conference on Machine Learning*, pages 242–252. PMLR, 2019b.

Md Zahangir Alom, Chris Yakopcic, Mst Nasrin, Tarek M Taha, Vijayan K Asari, et al. Breast cancer classification from histopathological images with inception recurrent residual convolutional neural network. *Journal of digital imaging*, 2019.

Setlur Amrith, Saurabh Garg, and Sergey Smith, Virginiaand Levine. Prompting is a double-edged sword: Improving worst-group robustness of foundation models. In *International Conference on Machine Learning*, 2024.

Jing An, Lexing Ying, and Yuhua Zhu. Why resampling outperforms reweighting for correcting sampling bias with stochastic gradients. *arXiv preprint arXiv:2009.13447*, 2020.

Sanjeev Arora, Rong Ge, Behnam Neyshabur, and Yi Zhang. Stronger generalization bounds for deep nets via a compression approach. *arXiv preprint arXiv:1802.05296*, 2018.

Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International Conference on Machine Learning (ICML)*, 2019a.

Sanjeev Arora, Hrishikesh Khandeparkar, Mikhail Khodak, Orestis Plevrakis, and Nikunj Saunshi. A theoretical analysis of contrastive unsupervised representation learning. *arXiv preprint arXiv:1902.09229*, 2019b.

Devansh Arpit, Stanislaw Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 233–242. JMLR. org, 2017.

Jordan Ash and Ryan P Adams. On warm-starting neural network training. *Advances in neural information processing systems*, 33:3884–3894, 2020.

Kamyar Azizzadenesheli, Anqi Liu, Fanny Yang, and Animashree Anandkumar. Regularized learning for domain adaptation under label shifts. In *International Conference on Learning Representations (ICLR)*, 2019.

Dheeraj Baby and Yu-Xiang Wang. Online forecasting of total-variation-bounded sequences. In *Neural Information Processing Systems (NeurIPS)*, 2019.

Dheeraj Baby and Yu-Xiang Wang. Optimal dynamic regret in exp-concave online learning. In *COLT*, 2021.

Dheeraj Baby and Yu-Xiang Wang. Optimal dynamic regret in proper online learning with strongly convex losses and beyond. *AISTATS*, 2022.

Dheeraj Baby and Yu-Xiang Wang. Second order path variationals in non-stationary online learning. *AISTATS*, 2023.

Dheeraj Baby, Xuandong Zhao, and Yu-Xiang Wang. An optimal reduction of tv-denoising to adaptive online learning. *AISTATS*, 2021.

Dheeraj Baby, Saurabh Garg, Tzu-Ching Yen, Sivaraman Balakrishnan, Zachary Chase

Lipton, and Yu-Xiang Wang. Online label shift: Optimal dynamic regret meets practical algorithms. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Christina Baek, Yiding Jiang, Aditi Raghunathan, and Zico Kolter. Agreement-on-the-line: Predicting the performance of neural networks under distribution shift. *arXiv preprint arXiv:2206.13089*, 2022.

Yong Bai, Yu-Jie Zhang, Peng Zhao, Masashi Sugiyama, and Zhi-Hua Zhou. Adapting to online label shift with provable guarantees. In *Advances in Neural Information Processing Systems*, 2022.

Mahsa Baktashmotlagh, Masoud Faraki, Tom Drummond, and Mathieu Salzmann. Learning factorized representations for open-set domain adaptation. In *International Conference on Learning Representations (ICLR)*, 2019.

Yogesh Balaji, Mehrdad Farajtabar, Dong Yin, Alex Mott, and Ang Li. The effectiveness of memory replay in large scale continual learning. *arXiv preprint arXiv:2010.02418*, 2020.

Peter Bandi, Oscar Geessink, Quirine Manson, Marcory Van Dijk, Maschenka Balkenhol, Meyke Hermsen, Babak Ehteshami Bejnordi, Byungjae Lee, Kyunghyun Paeng, Aoxiao Zhong, et al. From detection of individual metastases to classification of lymph node status at the patient level: the camelyon17 challenge. *IEEE Transactions on Medical Imaging*, 2018.

Yamini Bansal, Gal Kaplun, and Boaz Barak. For self-supervised learning, rationality implies generalization, provably. *arXiv preprint arXiv:2010.08508*, 2020.

Yujia Bao and Regina Barzilay. Learning to split for automatic bias detection. *arXiv preprint arXiv:2204.13749*, 2022.

Andrei Barbu, David Mayo, Julian Alverio, William Luo, Christopher Wang, Dan Gutfreund, Josh Tenenbaum, and Boris Katz. Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32. Curran Associates, Inc., 2019. https://proceedings.neurips.cc/paper/2019/file/97af07a14cacba681feacf3012730892-Paper.pdf.

Rémi Bardenet, Odalric-Ambrym Maillard, et al. Concentration inequalities for sampling without replacement. *Bernoulli*, 21(3):1361–1385, 2015.

Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. *arXiv preprint arXiv:2105.04906*, 2021.

Árpád Baricz. Mills' ratio: Monotonicity patterns and functional inequalities. *Journal of Mathematical Analysis and Applications*, 340(2):1362–1370, 2008.

Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky. Spectrally-normalized margin bounds for neural networks. In *Advances in neural information processing systems*, pages

6240–6249, 2017.

Sara Beery, Elijah Cole, and Arvi Gjoka. The iwildcam 2020 competition dataset. *arXiv preprint arXiv:2004.10340*, 2020.

Jessa Bekker and Jesse Davis. Estimating the class prior in positive and unlabeled data through decision tree induction. In *Assosication for the Advancement of Artificial Intelligence (AAAI)*, 2018.

Jessa Bekker and Jesse Davis. Learning from positive and unlabeled data: a survey. *Machine Learning*, 2020.

Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. In *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2006.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79(1-2), 2010a.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79:151–175, 2010b.

Shai Ben-David, Tyler Lu, Teresa Luu, and Dávid Pál. Impossibility Theorems for Domain Adaptation. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010c.

Aharon Ben-Tal, Dick Den Hertog, Anja De Waegenaere, Bertrand Melenberg, and Gijs Rennen. Robust solutions of optimization problems affected by uncertain probabilities. *Management Science*, 59(2):341–357, 2013.

Abhijit Bendale and Terrance Boult. Towards open world recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1893–1902, 2015.

David Berthelot, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. *arXiv preprint arXiv:1911.09785*, 2019.

David Berthelot, Rebecca Roelofs, Kihyuk Sohn, Nicholas Carlini, and Alex Kurakin. Adamatch: A unified approach to semi-supervised learning and domain adaptation. *arXiv preprint arXiv:2106.04732*, 2021.

Omar Besbes, Yonatan Gur, and Assaf Zeevi. Non-stationary stochastic optimization. *Operations research*, 63(5):1227–1244, 2015.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR, 2023.

Christopher M Bishop. *Pattern Recognition and Machine Learning.* Springer, 2006.

Gilles Blanchard, Gyemin Lee, and Clayton Scott. Semi-supervised novelty detection. *Journal of Machine Learning Research (JMLR)*, 11:2973–3009, 2010.

Gilles Blanchard, Gyemin Lee, and Clayton Scott. Generalizing from several related classification tasks to a new unlabeled sample. *Advances in neural information processing systems*, 24, 2011.

Jose Blanchet and Karthyek Murthy. Quantifying distributional model risk via optimal transport. *Mathematics of Operations Research*, 44(2):565–600, 2019.

Avrim Blum and Moritz Hardt. The ladder: A reliable leaderboard for machine learning competitions. In *International Conference on Machine Learning*, pages 1006–1014. PMLR, 2015.

Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K Warmuth. Learnability and the vapnik-chervonenkis dimension. *Journal of the ACM (JACM)*, 36(4): 929–965, 1989.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Daniel Borkan, Lucas Dixon, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. Nuanced metrics for measuring unintended bias with real data for text classification. In *Companion Proceedings of The 2019 World Wide Web Conference*, 2019.

Jorg Bornschein, Alexandre Galashov, Ross Hemsley, Amal Rannen-Triki, Yutian Chen, Arslan Chaudhry, Xu Owen He, Arthur Douillard, Massimo Caccia, Qixuang Feng, et al. Nevis'22: A stream of 100 tasks sampled from 30 years of computer vision research. *arXiv preprint arXiv:2211.11747*, 2022.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *European Conference on Computer Vision (ECCV)*, 2014. https://link.springer.com/chapter/10.1007/978-3-319-10599-4_29.

Olivier Bousquet and André Elisseeff. Stability and generalization. *Journal of machine learning research*, 2(Mar):499–526, 2002.

Olivier Bousquet, Stéphane Boucheron, and Gábor Lugosi. Introduction to statistical learning theory. In *Advanced Lectures on Machine Learning*, Lecture Notes in Computer Science. Springer, 2003.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Silvia Bucci, Mohammad Reza Loghmani, and Tatiana Tommasi. On the effectiveness of image rotation for open set domain adaptation. In *European Conference on Computer Vision*. Springer, 2020.

Mateusz Buda, Atsuto Maki, and Maciej A Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural networks*, 106:249–259, 2018.

Jonathon Byrd and Zachary C Lipton. What is the effect of importance weighting in deep learning? In *International Conference on Machine Learning (ICML)*, 2019.

Vivien Cabannes, Bobak T Kiani, Randall Balestriero, Yann LeCun, and Alberto Bietti. The ssl interplay: Augmentations, inductive bias, and generalization. *arXiv preprint arXiv:2302.02774*, 2023.

Tianle Cai, Ruiqi Gao, Jason Lee, and Qi Lei. A theory of label propagation for subpopulation shift. In *International Conference on Machine Learning*, pages 1170–1182. PMLR, 2021a.

Zhipeng Cai, Ozan Sener, and Vladlen Koltun. Online continual learning with natural distribution shifts: An empirical study with visual data. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8281–8290, 2021b.

Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. In *Advances in Neural Information Processing Systems*, volume 32, 2019a.

Kaidi Cao, Maria Brbic, and Jure Leskovec. Concept learners for few-shot learning. In *International Conference on Learning Representations (ICLR)*, 2021.

Kaidi Cao, Maria Brbic, and Jure Leskovec. Open-world semi-supervised learning. In *International Conference on Learning Representations (ICLR)*, 2022.

Zhangjie Cao, Kaichao You, Mingsheng Long, Jianmin Wang, and Qiang Yang. Learning to transfer examples for partial domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2985–2994, 2019b.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, 33:9912–9924, 2020.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

Fabio Cermelli, Massimiliano Mancini, Samuel Rota Bulo, Elisa Ricci, and Barbara Caputo. Modeling the background for incremental learning in semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9233–9242, 2020.

Nicolo Cesa-Bianchi and Gabor Lugosi. *Prediction, Learning, and Games.* Cambridge University Press, New York, NY, USA, 2006. ISBN 0521841089.

Ting-Jui Chang and Shahin Shahrampour. On online optimization: Dynamic regret analysis of strongly convex and smooth problems. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning. 2006. *Cambridge, Massachusettes: The MIT Press View Article*, 2, 2006.

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny.

Efficient lifelong learning with a-gem. *arXiv preprint arXiv:1812.00420*, 2018.

Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*, 2019.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16: 321–357, 2002.

Jiefeng Chen, Frederick Liu, Besim Avci, Xi Wu, Yingyu Liang, and Somesh Jha. Detecting errors and estimating accuracy on unlabeled data with self-training ensembles. *Advances in Neural Information Processing Systems*, 34:14980–14992, 2021a.

Mayee Chen, Karan Goel, Nimit S Sohoni, Fait Poms, Kayvon Fatahalian, and Christopher Ré. Mandoline: Model evaluation under distribution shift. In *International Conference on Machine Learning*, pages 1617–1629. PMLR, 2021b.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020a.

Xi Chen, Yining Wang, and Yu-Xiang Wang. Technical note — non-stationary stochastic optimization under lp, q-variation measures. *Operations Research*, 2018.

Xuxi Chen, Wuyang Chen, Tianlong Chen, Ye Yuan, Chen Gong, Kewei Chen, and Zhangyang Wang. Self-pu: Self boosted and calibrated positive-unlabeled training. In *International Conference on Machine Learning*, pages 1510–1519. PMLR, 2020b.

Zixiang Chen, Yihe Deng, Yuanzhi Li, and Quanquan Gu. Understanding transferable representation learning and zero-shot transfer in clip. *arXiv preprint arXiv:2310.00927*, 2023.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. Self-play fine-tuning converts weak language models to strong language models. *arXiv preprint arXiv:2401.01335*, 2024.

Gong Cheng, Junwei Han, and Xiaoqiang Lu. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the Institute of Electrical and Electronics Engineers (IEEE)*, 2017. https://ieeexplore.ieee.org/abstract/document/7891544.

Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. Reproducible scaling laws for contrastive language-image learning, 2022. https://arxiv.org/abs/2212.07143.

Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. Reproducible scaling laws for contrastive language-image learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2818–2829, 2023.

Lenaic Chizat and Francis Bach. Implicit bias of gradient descent for wide two-layer neural networks trained with the logistic loss. In *Conference on Learning Theory*, pages 1305–1338. PMLR, 2020.

Lenaic Chizat, Edouard Oyallon, and Francis Bach. On lazy training in differentiable programming. In *Advances in Neural Information Processing Systems*, pages 2937–2947, 2019.

Gordon Christie, Neil Fendley, James Wilson, and Ryan Mukherjee. Functional map of the world. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

Ching-Yao Chuang, Antonio Torralba, and Stefanie Jegelka. Estimating generalization under distribution shifts via domain-invariant representations. *arXiv preprint arXiv:2007.03511*, 2020.

Colin B Clement, Matthew Bierbaum, Kevin P O'Keeffe, and Alexander A Alemi. On the use of arxiv as a dataset. *arXiv preprint arXiv:1905.00075*, 2019.

Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011. https://proceedings.mlr.press/v15/coates11a.html.

Tabula Muris Consortium et al. A single cell transcriptomic atlas characterizes aging tissues in the mouse. *Nature*, 583(7817), 2020.

Corinna Cortes and Mehryar Mohri. Domain adaptation and sample bias correction theory and algorithm for regression. *Theoretical Computer Science*, 519, 2014.

Corinna Cortes, Yishay Mansour, and Mehryar Mohri. Learning Bounds for Importance Weighting. In *Advances in Neural Information Processing Systems (NIPS)*, 2010.

Andrea Cossu, Gabriele Graffieti, Lorenzo Pellegrini, Davide Maltoni, Davide Bacciu, Antonio Carta, and Vincenzo Lomonaco. Is class-incremental enough for continual learning? *Frontiers in Artificial Intelligence*, 5:829842, 2022.

Elliot Creager, Jörn-Henrik Jacobsen, and Richard Zemel. Environment inference for invariant learning. In *International Conference on Machine Learning*, pages 2189–2200. PMLR, 2021.

Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020.

Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9268–9277, 2019.

Amit Daniely, Alon Gonen, and Shai Shalev-Shwartz. Strongly adaptive online learning. In *International Conference on Machine Learning*, pages 1405–1411, 2015.

Luke N Darlow, Elliot J Crowley, Antreas Antoniou, and Amos J Storkey. Cinic-10 is not imagenet or cifar-10. *arXiv preprint arXiv:1810.03505*, 2018.

Francesco De Comité, François Denis, Rémi Gilleron, and Fabien Letouzey. Positive and unlabeled examples help learning. In *International Conference on Algorithmic Learning Theory (ALT)*. Springer, 1999.

Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2021.

Charles-Alban Deledalle, Loic Denis, Sonia Tabti, and Florence Tupin. *Closed-form expressions of the eigen decomposition of 2 x 2 and 3 x 3 Hermitian matrices*. PhD thesis, Université de Lyon, 2017.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *Computer Vision and Pattern Recognition (CVPR)*, 2009.

Weijian Deng and Liang Zheng. Are labels always necessary for classifier accuracy evaluation? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15069–15078, 2021.

Weijian Deng, Stephen Gould, and Liang Zheng. What does rotation prediction tell us about classifier accuracy under varying testing environments? *arXiv preprint arXiv:2106.05961*, 2021.

François Denis. Pac learning from positive statistical queries. In *International Conference on Algorithmic Learning Theory (ALT)*. Springer, 1998.

Karan Desai, Gaurav Kaul, Zubin Aysola, and Justin Johnson. Redcaps: Web-curated image-text data created by the people, for the people. *arXiv preprint arXiv:2111.11431*, 2021.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics (NAACL)*, 2019.

Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

Natalia Díaz-Rodríguez, Vincenzo Lomonaco, David Filliat, and Davide Maltoni. Don't forget, there is more than forgetting: new metrics for continual learning. *arXiv preprint arXiv:1810.13166*, 2018.

Frances Ding, Moritz Hardt, John Miller, and Ludwig Schmidt. Retiring adult: New datasets for fair machine learning. *Advances in Neural Information Processing Systems*, 34:6478–6490, 2021.

Yuxuan Ding, Lingqiao Liu, Chunna Tian, Jingyuan Yang, and Haoxuan Ding. Don't stop learning: Towards continual learning for the clip model. *arXiv preprint arXiv:2207.09248*, 2022.

Josip Djolonga, Jessica Yung, Michael Tschannen, Rob Romijnders, Lucas Beyer, Alexander Kolesnikov, Joan Puigcerver, Matthias Minderer, Alexander D'Amour, Dan Moldovan, et al. On robustness and transferability of convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16458–16468, 2021.

Shibhansh Dohare, Juan Hernandez-Garcia, Parash Rahman, Richard Sutton, and A Rupam Mahmood. Loss of plasticity in deep continual learning. 2023.

David Donoho and Iain Johnstone. Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, 81(3):425–455, 1994a.

David Donoho and Iain Johnstone. Minimax risk over ℓ_p-balls for ℓ_q-error. *Probability Theory and Related Fields*, 99(2):277–303, 1994b.

David Donoho, Richard Liu, and Brenda MacGibbon. Minimax risk over hyperrectangles, and implications. *Annals of Statistics*, 18(3):1416–1437, 1990.

David L Donoho and Iain M Johnstone. Minimax estimation via wavelet shrinkage. *The annals of Statistics*, 26(3):879–921, 1998.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021. https://openreview.net/forum?id=YicbFdNTTy.

Simon Du, Jason Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai. Gradient descent finds global minima of deep neural networks. In *International Conference on Machine Learning*, pages 1675–1685. PMLR, 2019.

Simon S Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. *arXiv preprint arXiv:1810.02054*, 2018.

Marthinus Du Plessis, Gang Niu, and Masashi Sugiyama. Convex formulation for learning from positive and unlabeled data. In *International conference on machine learning*, pages 1386–1394, 2015.

Marthinus C Du Plessis, Gang Niu, and Masashi Sugiyama. Analysis of learning from positive and unlabeled data. *Advances in neural information processing systems*, 27: 703–711, 2014.

Marthinus Christoffel Du Plessis and Masashi Sugiyama. Class prior estimation from positive and unlabeled data. *IEICE TRANSACTIONS on Information and Systems*, 97 (5):1358–1362, 2014a.

Marthinus Christoffel Du Plessis and Masashi Sugiyama. Semi-supervised learning of class balance under class-prior change by distribution matching. *Neural Networks*, 50:110–119, 2014b.

John Duchi, Peter Glynn, and Hongseok Namkoong. Statistics of robust optimization: A generalized empirical likelihood approach. *arXiv preprint arXiv:1610.03425*, 2016.

John C. Duchi and Hongseok Namkoong. Learning models with uniform performance via distributionally robust optimization. *The Annals of Statistics*, 49(3):1378 – 1406, 2021. doi: 10.1214/20-AOS2004. URL https://doi.org/10.1214/20-AOS2004.

Aryeh Dvoretzky, Jack Kiefer, and Jacob Wolfowitz. Asymptotic minimax character of the sample distribution function and of the classical multinomial estimator. *The Annals of Mathematical Statistics*, pages 642–669, 1956.

Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Leon Roth. Preserving statistical validity in adaptive data analysis. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 117–126, 2015.

Gintare Karolina Dziugaite and Daniel M Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *arXiv preprint arXiv:1703.11008*, 2017.

Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

André Elisseeff, Massimiliano Pontil, et al. Leave-one-out error and stability of learning algorithms with applications. *NATO science series sub series iii computer and systems sciences*, 190:111–130, 2003.

Charles Elkan and Keith Noto. Learning classifiers from only positive and unlabeled data. In *International Conference Knowledge Discovery and Data Mining (KDD)*, pages 213–220, 2008.

M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results, 2007. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In *International Conference on Artificial Intelligence and Statistics*, pages 3762–3773. PMLR, 2020.

Jonathan Frankle, David J Schwab, and Ari S Morcos. The early phase of neural network training. *arXiv preprint arXiv:2002.10365*, 2020.

Jerome Friedman and Bogdan E Popescu. Gradient directed regularization for linear regression and classification. Technical report, Technical Report, Statistics Department, Stanford University, 2003.

Bo Fu, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. Learning to detect open classes for universal domain adaptation. In *European Conference on Computer Vision*. Springer, 2020.

Samir Yitzhak Gadre, Gabriel Ilharco, Alex Fang, Jonathan Hayase, Georgios Smyrnis, Thao Nguyen, Ryan Marten, Mitchell Wortsman, Dhruba Ghosh, Jieyu Zhang, et al. Datacomp: In search of the next generation of multimodal datasets. *arXiv preprint arXiv:2304.14108*, 2023.

Pierre Gaillard and Sébastien Gerchinovitz. A chaining algorithm for online nonparametric regression. In *Conference on Learning Theory*, pages 764–796, 2015.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The journal of machine learning research*, 2016.

Jacob Gardner, Geoff Pleiss, Kilian Q Weinberger, David Bindel, and Andrew G Wilson. Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Saurabh Garg, Yifan Wu, Sivaraman Balakrishnan, and Zachary Lipton. A unified view of label shift estimation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020a.

Saurabh Garg, Yifan Wu, Sivaraman Balakrishnan, and Zachary C Lipton. A unified view of label shift estimation. *arXiv preprint arXiv:2003.07554*, 2020b.

Saurabh Garg, Sivaraman Balakrishnan, Zico Kolter, and Zachary Lipton. RATT: Leveraging unlabeled data to guarantee generalization. In *International Conference on Machine Learning (ICML)*, 2021a.

Saurabh Garg, Yifan Wu, Alex Smola, Sivaraman Balakrishnan, and Zachary Lipton. Mixture proportion estimation and PU learning: A modern approach. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021b.

Saurabh Garg, Sivaraman Balakrishnan, and Zachary Lipton. Domain adaptation under open set label shift. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022a.

Saurabh Garg, Sivaraman Balakrishnan, Zachary Lipton, Behnam Neyshabur, and Hanie Sedghi. Leveraging unlabeled data to predict out-of-distribution performance. In *International Conference on Learning Representations (ICLR)*, 2022b.

Saurabh Garg, Sivaraman Balakrishnan, Zachary Chase Lipton, Behnam Neyshabur, and Hanie Sedghi. Leveraging unlabeled data to predict out-of-distribution performance. In *International Conference on Learning Representations*, 2022c.

Saurabh Garg, Nick Erickson, James Sharpnack, Alex Smola, Siva Balakrishnan, and Zachary Lipton. Rlsbench: A large-scale empirical study of domain adaptation under relaxed label shift. In *International Conference on Machine Learning (ICML)*, 2023a.

Saurabh Garg, Amrith Setlur, Zachary Lipton, Sivaraman Balakrishnan, Virginia Smith, and Aditi Raghunathan. Complementary benefits of contrastive learning and self-training under distribution shift. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023b.

Saurabh Garg, Mehrdad Farajtabar, Hadi Pouransari, Raviteja Vemulapalli, Sachin Mehta, Oncel Tuzel, Vaishaal Shankar, and Fartash Faghri. Tic-clip: Continual training of clip models. In *Internation Conference on Learning Representations*, 2024.

Quentin Garrido, Yubei Chen, Adrien Bardes, Laurent Najman, and Yann Lecun. On the duality between contrastive and non-contrastive self-supervised learning. *arXiv preprint* *arXiv:2206.02574*, 2022.

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. *arXiv preprint arXiv:1705.08500*, 2017.

Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. https://ieeexplore.ieee.org/abstract/document/6248074.

Hristijan Gjoreski, Mathias Ciliberto, Lin Wang, Francisco Javier Ordonez Morales, Sami Mekki, Stefan Valentin, and Daniel Roggen. The university of sussex-huawei locomotion and transportation dataset for multimodal analytics with mobile devices. *IEEE Access*, 6:42592–42604, 2018.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Artificial Intelligence and Statistics (AISTATS)*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

Gautam Goel and Adam Wierman. An online algorithm for smoothed regression and lqr control. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, 2019.

Ian J Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*, 2013.

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

Yves Grandvalet and Yoshua Bengio. Entropy regularization., 2006.

Arthur Gretton, Alexander J Smola, Jiayuan Huang, Marcel Schmittfull, Karsten M Borgwardt, and Bernhard Schölkopf. Covariate Shift by Kernel Mean Matching. *Journal of Machine Learning Research (JMLR)*, 2009.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

Devin Guillory, Vaishaal Shankar, Sayna Ebrahimi, Trevor Darrell, and Ludwig Schmidt. Predicting with confidence on unseen distributions. *arXiv preprint arXiv:2107.03315*, 2021.

Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. *arXiv preprint arXiv:2007.01434*, 2020.

Suriya Gunasekar, Jason Lee, Daniel Soudry, and Nathan Srebro. Implicit bias of gradient descent on linear convolutional networks. *arXiv preprint arXiv:1806.00468*, 2018a.

Suriya Gunasekar, Blake Woodworth, Srinadh Bhojanapalli, Behnam Neyshabur, and Nathan Srebro. Implicit regularization in matrix factorization. In *2018 Information Theory and Applications Workshop (ITA)*, pages 1–10. IEEE, 2018b.

Adityanand Guntuboyina, Donovan Lieu, Sabyasachi Chatterjee, and Bodhisattva Sen. Adaptive risk bounds in univariate total variation denoising and trend filtering. *Annals of Statistics*, 2020.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning (ICML)*, 2017.

Guy Gur-Ari, Daniel A Roberts, and Ethan Dyer. Gradient descent happens in a tiny subspace. *arXiv preprint arXiv:1812.04754*, 2018.

Raia Hadsell, Dushyant Rao, Andrei A Rusu, and Razvan Pascanu. Embracing change: Continual learning in deep neural networks. *Trends in cognitive sciences*, 24(12):1028–1040, 2020.

Melissa Hall, Laura Gustafson, Aaron Adcock, Ishan Misra, and Candace Ross. Vision-language models performing zero-shot tasks exhibit disparities between gender groups. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2778–2785, 2023.

Jeff Z HaoChen and Tengyu Ma. A theoretical study of inductive biases in contrastive learning. *arXiv preprint arXiv:2211.14699*, 2022.

Jeff Z HaoChen, Colin Wei, Adrien Gaidon, and Tengyu Ma. Provable guarantees for self-supervised deep learning with spectral contrastive loss. *Advances in Neural Information Processing Systems*, 34:5000–5011, 2021.

Jeff Z HaoChen, Colin Wei, Ananya Kumar, and Tengyu Ma. Beyond separability: Analyzing the linear transferability of contrastive representations to related subpopulations. *arXiv preprint arXiv:2204.02683*, 2022.

Moritz Hardt, Ben Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *International Conference on Machine Learning*, pages 1225–1234. PMLR, 2016.

Tyler L Hayes, Nathan D Cahill, and Christopher Kanan. Memory efficient experience replay for streaming learning. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 9769–9776. IEEE, 2019.

Elad Hazan. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016.

Elad Hazan and Comandur Seshadhri. Adaptive algorithms for online decision problems. In *Electronic colloquium on computational complexity (ECCC)*, volume 14, 2007.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Computer Vision and Pattern Recognition (CVPR)*, 2016.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

James J Heckman. Sample Selection Bias as a Specification Error (With an Application to the Estimation of Labor Supply Functions), 1977.

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7): 2217–2226, 2019.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

Dan Hendrycks and Kevin Gimpel. A Baseline for Detecting Misclassified and Out-Of-Distribution Examples in Neural Networks. In *International Conference on Learning Representations (ICLR)*, 2017.

Dan Hendrycks, Steven Basart, Mantas Mazeika, Mohammadreza Mostajabi, Jacob Steinhardt, and Dawn Song. Scaling out-of-distribution detection for real-world settings. *arXiv preprint arXiv:1911.11132*, 2019.

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. *ICCV*, 2021a. https://arxiv.org/abs/2006.16241.

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8340–8349, 2021b.

Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021c. https://arxiv.org/abs/1907.07174.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. In *The Collected Works of Wassily Hoeffding*, pages 409–426. Springer, 1994.

Yen-Chang Hsu, Yen-Cheng Liu, Anita Ramasamy, and Zsolt Kira. Re-evaluating continual learning scenarios: A categorization and case for strong baselines. *arXiv preprint arXiv:1810.12488*, 2018.

Huiyi Hu, Ang Li, Daniele Calandriello, and Dilan Gorur. One pass imagenet. *arXiv preprint arXiv:2111.01956*, 2021.

Wei Hu, Zhiyuan Li, and Dingli Yu. Simple and effective regularization methods for training on noisily labeled data with generalization guarantee. *arXiv preprint arXiv:1905.11368*, 2019.

Wei Hu, Lechao Xiao, Ben Adlam, and Jeffrey Pennington. The surprising simplicity of the early-time learning dynamics of neural networks. *arXiv preprint arXiv:2006.14599*, 2020.

Weihua Hu, Gang Niu, Issei Sato, and Masashi Sugiyama. Does distributionally robust supervised learning give robust classifiers? In *International Conference on Machine Learning*, pages 2029–2037. PMLR, 2018.

Chen Huang, Yining Li, Chen Change Loy, and Xiaoou Tang. Learning deep representation for imbalanced classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5375–5384, 2016.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

Like Hui and Mikhail Belkin. Evaluation of neural architectures trained with square loss vs cross-entropy in classification tasks. *arXiv preprint arXiv:2006.07322*, 2020.

Jonathan J. Hull. A database for handwritten text recognition research. *IEEE Transactions on pattern analysis and machine intelligence*, 16(5):550–554, 1994.

Badr Youbi Idrissi, Martin Arjovsky, Mohammad Pezeshki, and David Lopez-Paz. Simple data balancing achieves competitive worst-group-accuracy. In *Conference on Causal Learning and Reasoning*, pages 336–351. PMLR, 2022.

Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, July 2021. URL https://doi.org/10.5281/zenodo.5143773. If you use this software, please cite it as below.

Gabriel Ilharco, Mitchell Wortsman, Samir Yitzhak Gadre, Shuran Song, Hannaneh Hajishirzi, Simon Kornblith, Ali Farhadi, and Ludwig Schmidt. Patching open-vocabulary models by interpolating weights. *Advances in Neural Information Processing Systems*, 35:29262–29277, 2022.

Dmitry Ivanov. DEDPUL: Difference-of-estimated-densities-based positive-unlabeled learning. *arXiv preprint arXiv:1902.06965*, 2019.

Andrew Jacobsen and Ashok Cutkosky. Parameter-free mirror descent. *COLT*, 2022.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, pages 8571–8580, 2018.

Ali Jadbabaie, Alexander Rakhlin, Shahin Shahrampour, and Karthik Sridharan. Online optimization: Competing with dynamic comparators. In *Artificial Intelligence and Statistics*, pages 398–406, 2015.

Shantanu Jain, Martha White, Michael W Trosset, and Predrag Radivojac. Nonparametric semi-supervised learning of class proportions. *arXiv preprint arXiv:1601.01944*, 2016.

Joel Jang, Seonghyeon Ye, Sohee Yang, Joongbo Shin, Janghoon Han, Gyeonghun Kim, Stanley Jungkyu Choi, and Minjoon Seo. Towards continual knowledge learning of language models. *arXiv preprint arXiv:2110.03215*, 2021.

Joel Jang, Seonghyeon Ye, Changho Lee, Sohee Yang, Joongbo Shin, Janghoon Han, Gyeonghun Kim, and Minjoon Seo. Temporalwiki: A lifelong benchmark for training and evaluating ever-evolving language models. *arXiv preprint arXiv:2204.14211*, 2022.

Xu Ji, Razvan Pascanu, Devon Hjelm, Andrea Vedaldi, Balaji Lakshminarayanan, and Yoshua Bengio. Predicting unreliable predictions by shattering a neural network. *arXiv* *preprint arXiv:2106.08365*, 2021.

Ziwei Ji and Matus Telgarsky. The implicit bias of gradient descent on nonseparable data. In *Conference on Learning Theory*, pages 1772–1798. PMLR, 2019.

Heinrich Jiang, Been Kim, Melody Y Guan, and Maya R Gupta. To trust or not to trust a classifier. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5546–5557, 2018.

Junguang Jiang, Yang Shu, Jianmin Wang, and Mingsheng Long. Transferability in deep learning: A survey, 2022.

Yiding Jiang, Vaishnavh Nagarajan, Christina Baek, and J Zico Kolter. Assessing generalization of sgd via disagreement. *arXiv preprint arXiv:2106.13799*, 2021.

Thorsten Joachims et al. Transductive inference for text classification using support vector machines. In *Icml*, volume 99, pages 200–209, 1999.

Fredrik D. Johansson, David Sontag, and Rajesh Ranganath. Support and invertibility in domain-invariant representations. In Kamalika Chaudhuri and Masashi Sugiyama, editors, *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, Proceedings of Machine Learning Research. PMLR, 2019.

Alistair Johnson, Lucas Bulgarelli, Tom Pollard, Steven Horng, Leo Anthony Celi, and Roger Mark. Mimic-iv. *PhysioNet. Available online at: https://physionet.org/content/mimiciv/1.0/(accessed August 23, 2021)*, 2020.

Daniel D Johnson, Ayoub El Hanchi, and Chris J Maddison. Contrastive learning can find an optimal basis for approximately view-invariant functions. *arXiv preprint arXiv:2210.01883*, 2022.

Iain M. Johnstone. Gaussian estimation: Sequence and wavelet models, 2017. URL: https://imjohnstone.su.domains//GE12-27-11.pdf. Last visited on 2023/04/20.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431. Association for Computational Linguistics, April 2017.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

Sham M Kakade, Karthik Sridharan, and Ambuj Tewari. On the complexity of linear prediction: Risk bounds, margin bounds, and regularization. *Advances in neural information processing systems*, 21, 2008.

Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. In *International Conference on Learning Representations*, 2020.

Akash Kannan, Saurabh Garg, and Sivaraman Balakrishnan. On the impossibility and possibility of domain adaptation methods. *Under Review*, 2024.

Michael Kearns and Dana Ron. Algorithmic stability and sanity-check bounds for leave-one-out cross-validation. *Neural computation*, 11(6):1427–1453, 1999.

Seung-Jean Kim, Kwangmoo Koh, Stephen Boyd, and Dimitry Gorinevsky. ℓ_1 trend filtering. *SIAM Review*, 51(2):339–360, 2009.

Diederik P Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv Preprint arXiv:1412.6980*, 2014.

Polina Kirichenko, Pavel Izmailov, and Andrew Gordon Wilson. Last layer re-training is sufficient for robustness to spurious correlations. *arXiv preprint arXiv:2204.02937*, 2022.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Ryuichi Kiryo, Gang Niu, Marthinus C Du Plessis, and Masashi Sugiyama. Positive-unlabeled learning with non-negative risk estimator. In *Advances in neural information processing systems*, pages 1675–1685, 2017.

Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, Tony Lee, Etienne David, Ian Stavness, Wei Guo, Berton A. Earnshaw, Imran S. Haque, Sara Beery, Jure Leskovec, Anshul Kundaje, Emma Pierson, Sergey Levine, Chelsea Finn, and Percy Liang. WILDS: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning (ICML)*, 2021.

Wouter M Koolen, Alan Malek, Peter L Bartlett, and Yasin Abbasi. Minimax time series prediction. *Advances in Neural Information Processing Systems (NIPS'15)*, pages 2557–2565, 2015.

Wojciech Kotłowski, Wouter M. Koolen, and Alan Malek. Online isotonic regression. In *Annual Conference on Learning Theory (COLT-16)*, volume 49, pages 1165–1189. PMLR, 2016.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *International Conference on Computer Vision Workshops (ICML)*, 2013. https://www.cv-foundation.org/openaccess/content_iccv_workshops_2013/W19/html/Krause_3D_Object_Representations_2013_ICCV_paper.html.

Alex Krizhevsky and Geoffrey Hinton. Learning Multiple Layers of Features from Tiny Images. Technical report, Citeseer, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Frank R Kschischang. The complementary error function. *Online, April*, 2017.

Volodymyr Kuleshov and Percy S Liang. Calibrated structured prediction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2015.

Ananya Kumar, Percy S Liang, and Tengyu Ma. Verified uncertainty calibration. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Ananya Kumar, Tengyu Ma, and Percy Liang. Understanding self-training for gradual domain adaptation. In *International Conference on Machine Learning*, pages 5468–5479. PMLR, 2020.

Ananya Kumar, Aditi Raghunathan, Robbie Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. *arXiv preprint arXiv:2202.10054*, 2022a.

Ananya Kumar, Aditi Raghunathan, Robbie Matthew Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. In *International Conference on Learning Representations*, 2022b. URL https://openreview.net/forum?id=UYneFzXSJWh.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86, 1998.

Dong-Hyun Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 896, 2013.

Jason D Lee, Qi Lei, Nikunj Saunshi, and Jiacheng Zhuo. Predicting what you already know helps: Provable self-supervised learning. *Advances in Neural Information Processing Systems*, 34:309–323, 2021.

Sang-Woo Lee, Jin-Hwa Kim, Jaehyun Jun, Jung-Woo Ha, and Byoung-Tak Zhang. Overcoming catastrophic forgetting by incremental moment matching. *Advances in neural information processing systems*, 30, 2017.

Wee Sun Lee and Bing Liu. Learning with positive and unlabeled examples using weighted logistic regression. In *International Conference on Machine Learning (ICML)*, 2003.

Yoonho Lee, Huaxiu Yao, and Chelsea Finn. Diversify and disambiguate: Learning from underspecified data. *arXiv preprint arXiv:2202.03418*, 2022.

Yoonho Lee, Michelle Lam, Helena Vasconcelos, Michael Bernstein, and Chelsea Finn. Interactive model correction with natural language. In *XAI in Action: Past, Present, and Future Applications*, 2023.

Fabien Letouzey, François Denis, and Rémi Gilleron. Learning from positive and unlabeled examples. In *International Conference on Algorithmic Learning Theory*, pages 71–85. Springer, 2000.

Mingchen Li, Mahdi Soltanolkotabi, and Samet Oymak. Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. *arXiv preprint arXiv:1903.11680*, 2019.

Mingchen Li, Mahdi Soltanolkotabi, and Samet Oymak. Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 4313–4324. PMLR, 2020.

Xiaoli Li and Bing Liu. Learning to classify texts using positive and unlabeled data. In *International Joint Conference on Artificial Intelligence (IJCAI)*. Citeseer, 2003.

Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. Revisiting batch normalization for practical domain adaptation. *arXiv preprint arXiv:1603.04779*, 2016.

Yuanzhi Li and Yingyu Liang. Learning overparameterized neural networks via stochastic gradient descent on structured data. In *Advances in Neural Information Processing Systems*, pages 8157–8166, 2018.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

Qing Lian, Wen Li, Lin Chen, and Lixin Duan. Known-class aware self-ensemble for open set domain adaptation. *arXiv preprint arXiv:1905.01068*, 2019.

Shiyu Liang, Yixuan Li, and R Srikant. Enhancing the Reliability of Out-Of-Distribution Image Detection in Neural Networks. In *International Conference on Learning Representations (ICLR)*, 2018.

Haofu Liao. A deep learning approach to universal skin disease classification. *University of Rochester Department of Computer Science, CSC*, 2016.

Zhiqiu Lin, Jia Shi, Deepak Pathak, and Deva Ramanan. The clear benchmark: Continual learning on real-world imagery. In *Thirty-fifth conference on neural information processing systems datasets and benchmarks track (round 2)*, 2021.

Zachary Lipton, Yu-Xiang Wang, and Alex Smola. Detecting and correcting for label shift with black box predictors. In *International Conference on Machine Learning*, 2018a.

Zachary C Lipton, Yu-Xiang Wang, and Alex Smola. Detecting and Correcting for Label Shift with Black Box Predictors. In *International Conference on Machine Learning (ICML)*, 2018b.

Adam Liška, Tomáš Kočiský, Elena Gribovskaya, Tayfun Terzi, Eren Sezener, Devang Agrawal, Cyprien de Masson d'Autume, Tim Scholtes, Manzil Zaheer, Susannah Young, Ellen Gilsenan-McMahon Sophia Austin, Phil Blunsom, and Angeliki Lazaridou. Streamingqa: A benchmark for adaptation to new knowledge over time in question answering models. *arXiv preprint arXiv:2205.11388*, 2022.

Bing Liu, Wee Sun Lee, Philip S Yu, and Xiaoli Li. Partially supervised classification of text documents. In *International Conference on Machine Learning (ICML)*, 2002.

Bing Liu, Yang Dai, Xiaoli Li, Wee Sun Lee, and Philip S Yu. Building text classifiers using positive and unlabeled examples. In *International Conference on Data Mining (ICDM)*, 2003.

Evan Z Liu, Behzad Haghgoo, Annie S Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. Just train twice: Improving group robustness without training group information. In *International Conference on Machine Learning*, pages 6781–6792. PMLR, 2021a.

Hong Liu, Zhangjie Cao, Mingsheng Long, Jianmin Wang, and Qiang Yang. Separate to adapt: Open set domain adaptation via progressive separation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2927–2936, 2019a.

Sheng Liu, Jonathan Niles-Weed, Narges Razavian, and Carlos Fernandez-Granda. Early-learning regularization prevents memorization of noisy labels. *arXiv preprint arXiv:2007.00151*, 2020.

Xiaofeng Liu, Zhenhua Guo, Site Li, Fangxu Xing, Jane You, C.-C. Jay Kuo, Georges El Fakhri, and Jonghye Woo. Adversarial Unsupervised Domain Adaptation with Conditional and Label Shift: Infer, Align and Iterate. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10347–10356, Montreal, QC, Canada, October 2021b. IEEE. ISBN 978-1-66542-812-5. doi: 10.1109/ICCV48922.2021.01020. URL https://ieeexplore.ieee.org/document/9710205/.

Xingyu Liu, Alex Leonardi, Lu Yu, Chris Gilmer-Hill, Matthew Leavitt, and Jonathan Frankle. Knowledge distillation for efficient sequences of training runs. *arXiv preprint arXiv:2303.06480*, 2023.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019b.

Vincenzo Lomonaco and Davide Maltoni. Core50: a new dataset and benchmark for continuous object recognition. In *Conference on robot learning*, pages 17–26. PMLR, 2017.

Vincenzo Lomonaco, Lorenzo Pellegrini, Pau Rodriguez, Massimo Caccia, Qi She, Yu Chen, Quentin Jodelet, Ruiping Wang, Zheda Mai, David Vazquez, et al. Cvpr 2020 continual learning in computer vision competition: Approaches, results, current challenges and future directions. *Artificial Intelligence*, 303:103635, 2022.

Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *International conference on machine learning*, pages 97–105. PMLR, 2015.

Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Deep transfer learning with joint adaptation networks. In *International conference on machine learning*. PMLR, 2017.

Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. *Advances in neural information processing systems*, 31, 2018.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30, 2017.

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Yuzhe Lu, Zhenlin Wang, Runtian Zhai, Soheil Kolouri, Joseph Campbell, and Katia Sycara. Predicting out-of-distribution error with confidence optimal transport. In *ICLR 2023 Workshop on Pitfalls of limited data and computation for Trustworthy ML*, 2023.

Martin Q Ma, Yao-Hung Hubert Tsai, Paul Pu Liang, Han Zhao, Kun Zhang, Ruslan Salakhutdinov, and Louis-Philippe Morency. Conditional contrastive learning for improving fairness in self-supervised learning. *arXiv preprint arXiv:2106.02866*, 2021.

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Association for Computational Linguistics (ACL)*, 2011.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Enno Mammen and Sara van de Geer. Locally adaptive regression splines. *Annals of Statistics*, 25(1):387–413, 1997.

Jeroen Manders, Twan van Laarhoven, and Elena Marchiori. Adversarial Alignment of Class Prediction Uncertainties for Domain Adaptation, January 2019. URL http://arxiv.org/abs/1804.04448. Number: arXiv:1804.04448 arXiv:1804.04448 [cs, stat].

Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation: Learning bounds and algorithms. *arXiv preprint arXiv:0902.3430*, 2009.

Colin McDiarmid. *On the method of bounded differences*, page 148–188. London Mathematical Society Lecture Note Series. Cambridge University Press, 1989.

Umberto Michieli and Pietro Zanuttigh. Incremental learning techniques for semantic segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision workshops*, pages 0–0, 2019.

John P Miller, Rohan Taori, Aditi Raghunathan, Shiori Sagawa, Pang Wei Koh, Vaishaal Shankar, Percy Liang, Yair Carmon, and Ludwig Schmidt. Accuracy on the line: on the strong correlation between out-of-distribution and in-distribution generalization. In *International Conference on Machine Learning*. PMLR, 2021.

Seyed Iman Mirzadeh, Mehrdad Farajtabar, and Hassan Ghasemzadeh. Dropout as an implicit gating mechanism for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 232–233, 2020a.

Seyed Iman Mirzadeh, Mehrdad Farajtabar, Razvan Pascanu, and Hassan Ghasemzadeh. Understanding the role of training regimes in continual learning. *Advances in Neural Information Processing Systems*, 33:7308–7320, 2020b.

Samarth Mishra, Kate Saenko, and Venkatesh Saligrama. Surprisingly simple semi-supervised domain adaptation with pretraining and consistency. *arXiv preprint arXiv:2101.12727*, 2021.

Jovana Mitrovic, Brian McWilliams, Jacob Walker, Lars Buesing, and Charles Blundell. Representation learning via invariant causal mechanisms. *arXiv preprint arXiv:2010.07922*, 2020.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. The MIT Press, 2012.

Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *International Conference on Machine Learning*, pages 10–18. PMLR, 2013.

Sayan Mukherjee, Partha Niyogi, Tomaso Poggio, and Ryan Rifkin. Learning theory: stability is sufficient for generalization and necessary and sufficient for consistency of empirical risk minimization. *Advances in Computational Mathematics*, 25(1):161–193, 2006.

Kevin P Murphy. *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

Vidya Muthukumar, Adhyyan Narang, Vignesh Subramanian, Mikhail Belkin, Daniel Hsu, and Anant Sahai. Classification vs regression in overparameterized regimes: Does the loss function matter? *arXiv preprint arXiv:2005.08054*, 2020.

Vaishnavh Nagarajan and J Zico Kolter. Deterministic pac-bayesian generalization bounds for deep networks via generalizing noise-resilience. *arXiv preprint arXiv:1905.13344*, 2019a.

Vaishnavh Nagarajan and J Zico Kolter. Uniform convergence may be unable to explain generalization in deep learning. In *Advances in Neural Information Processing Systems*, pages 11615–11626, 2019b.

Vaishnavh Nagarajan, Anders Andreassen, and Behnam Neyshabur. Understanding the failure modes of out-of-distribution generalization. *arXiv preprint arXiv:2010.15775*, 2020.

Preetum Nakkiran, Gal Kaplun, Dimitris Kalimeris, Tristan Yang, Benjamin L Edelman, Fred Zhang, and Boaz Barak. Sgd on neural networks learns functions of increasing complexity. *arXiv preprint arXiv:1905.11604*, 2019.

Junhyun Nam, Hyuntak Cha, Sungsoo Ahn, Jaeho Lee, and Jinwoo Shin. Learning from failure: De-biasing classifier from biased classifier. *Advances in Neural Information Processing Systems*, 33:20673–20684, 2020.

Jeffrey Negrea, Gintare Karolina Dziugaite, and Daniel Roy. In defense of uniform convergence: Generalization via derandomization with an application to interpolating predictors. In *International Conference on Machine Learning*, pages 7263–7272. PMLR, 2020.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading Digits in Natural Images With Unsupervised Feature Learning. 2011a.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *Advances in Neural Information Processing Systems (NeurIPS) Workshops*, 2011b. https://storage.googleapis.com/pub-tools-public-publication-data/pdf/37648.pdf.

Gergely Neu and Lorenzo Rosasco. Iterate averaging as regularization for stochastic gradient descent. In *Conference On Learning Theory*, pages 3222–3242. PMLR, 2018.

Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. Norm-based capacity control in neural networks. In *Conference on Learning Theory*, pages 1376–1401, 2015.

Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nathan Srebro. Exploring generalization in deep learning. *arXiv preprint arXiv:1706.08947*, 2017a.

Behnam Neyshabur, Srinadh Bhojanapalli, and Nathan Srebro. A pac-bayesian approach to spectrally-normalized margin bounds for neural networks. *arXiv preprint arXiv:1707.09564*, 2017b.

Behnam Neyshabur, Zhiyuan Li, Srinadh Bhojanapalli, Yann LeCun, and Nathan Srebro. The role of over-parametrization in generalization of neural networks. In *International Conference on Learning Representations*, 2018.

Jianmo Ni, Jiacheng Li, and Julian McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019.

Zixuan Ni, Longhui Wei, Siliang Tang, Yueting Zhuang, and Qi Tian. Continual vision-language representaion learning with off-diagonal information. *arXiv preprint arXiv:2305.07437*, 2023.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics and Image Processing*, 2008. https://ieeexplore.ieee.org/document/4756141.

Luke Oakden-Rayner, Jared Dunnmon, Gustavo Carneiro, and Christopher Ré. Hidden stratification causes clinically meaningful failures in machine learning for medical imaging. In *Proceedings of the ACM conference on health, inference, and learning*, pages 151–159, 2020.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Francesco Orabona. A modern introduction to online learning, 2019.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, David Sculley, Sebastian Nowozin, Joshua V Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Matteo Pagliardini, Martin Jaggi, François Fleuret, and Sai Praneeth Karimireddy. Agree to disagree: Diversity through disagreement for better transferability. In *The Eleventh International Conference on Learning Representations*, 2023.

Pau Panareda Busto and Juergen Gall. Open set domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 754–763, 2017.

Vardan Papyan, XY Han, and David L Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of*

Sciences, 117(40):24652–24663, 2020.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural networks*, 113:54–71, 2019.

Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. https://ieeexplore.ieee.org/document/6248092.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. Visda: The visual domain adaptation challenge, 2017.

Xingchao Peng, Ben Usman, Kuniaki Saito, Neela Kaushik, Judy Hoffman, and Kate Saenko. Syn2real: A new benchmark forsynthetic-to-real visual domain adaptation. *arXiv preprint arXiv:1806.09755*, 2018.

Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1406–1415, 2019.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *North American Chapter of the Association for Computational Linguistics (NAACL)*, 2018.

PhysioToolkit PhysioBank. Physionet: components of a new research resource for complex physiologic signals. *Circulation*, 101(23):e215–e220, 2000.

Bryan A Plummer, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the IEEE international conference on computer vision*, pages 2641–2649, 2015.

Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. Gdumb: A simple approach that questions our progress in continual learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 524–540. Springer, 2020.

Ameya Prabhu, Hasan Abed Al Kader Hammoud, Puneet K Dokania, Philip HS Torr, Ser-Nam Lim, Bernard Ghanem, and Adel Bibi. Computationally budgeted continual learning: What does matter? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3698–3707, 2023.

Viraj Prabhu, Shivam Khare, Deeksha Kartik, and Judy Hoffman. Sentry: Selective entropy optimization via committee consistency for unsupervised domain adaptation.

In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8558–8567, 2021.

Stephan Rabanser, Stephan Günnemann, and Zachary C Lipton. Failing loudly: An empirical study of methods for detecting dataset shift. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

Anant Raj, Pierre Gaillard, and Christophe Saad. Non-stationary online regression, 2020.

Alexander Rakhlin and Karthik Sridharan. Online non-parametric regression. In *Conference on Learning Theory*, pages 1232–1264, 2014.

Harish Ramaswamy, Clayton Scott, and Ambuj Tewari. Mixture proportion estimation via kernel embeddings of distributions. In *International conference on machine learning*, pages 2052–2060, 2016.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do cifar-10 classifiers generalize to cifar-10? *arXiv preprint arXiv:1806.00451*, 2018.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet classifiers generalize to ImageNet? In *International Conference on Machine Learning (ICML)*, 2019a. http://proceedings.mlr.press/v97/recht19a.html.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International Conference on Machine Learning*, pages 5389–5400. PMLR, 2019b.

Henry Reeve and Ata Kabán. Exploiting geometric structure in mixture proportion estimation with generalised blanchard-lee-scott estimators. In *Proceedings of the 30th International Conference on Algorithmic Learning Theory*, pages 682–699. PMLR, 2019.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019. URL http://arxiv.org/abs/1908.10084.

Manley Roberts, Pranav Mani, Saurabh Garg, and Zachary Lipton. Unsupervised learning under latent label shift. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Anthony Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 7(2):123–146, 1995.

David Rolnick, Andreas Veit, Serge Belongie, and Nir Shavit. Deep learning is robust to massive label noise. *arXiv preprint arXiv:1705.10694*, 2017.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

Elan Rosenfeld and Saurabh Garg. (almost) provable error bounds under distribution shift via disagreement discrepancy. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Elan Rosenfeld, Pradeep Ravikumar, and Andrej Risteski. Domain-adjusted regression or: Erm may already learn features sufficient for out-of-distribution generalization. *arXiv preprint arXiv:2202.06856*, 2022.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

Veeranjaneyulu Sadhanala, Yu-Xiang Wang, and Ryan Tibshirani. Total variation classes beyond 1d: Minimax rates, and the limitations of linear smoothers. *Advances in Neural Information Processing Systems (NIPS-16)*, 2016a.

Veeranjaneyulu Sadhanala, Yu-Xiang Wang, and Ryan J Tibshirani. Graph sparsification approaches for laplacian smoothing. In *AISTATS'16*, pages 1250–1259, 2016b.

Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *European conference on computer vision*, pages 213–226. Springer, 2010.

Marco Saerens, Patrice Latinne, and Christine Decaestecker. Adjusting the Outputs of a Classifier to New a Priori Probabilities: A Simple Procedure. *Neural Computation*, 2002.

Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019.

Shiori Sagawa, Aditi Raghunathan, Pang Wei Koh, and Percy Liang. An investigation of why overparameterization exacerbates spurious correlations. In *International Conference on Machine Learning*, pages 8346–8356. PMLR, 2020.

Shiori Sagawa, Pang Wei Koh, Tony Lee, Irena Gao, Sang Michael Xie, Kendrick Shen, Ananya Kumar, Weihua Hu, Michihiro Yasunaga, Henrik Marklund, Sara Beery, Etienne David, Ian Stavness, Wei Guo, Jure Leskovec, Kate Saenko, Tatsunori Hashimoto, Sergey Levine, Chelsea Finn, and Percy Liang. Extending the wilds benchmark for unsupervised adaptation. In *NeurIPS Workshop on Distribution Shifts*, 2021.

Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE conference* *on computer vision and pattern recognition*, 2018a.

Kuniaki Saito, Shohei Yamamoto, Yoshitaka Ushiku, and Tatsuya Harada. Open set domain adaptation by backpropagation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 153–168, 2018b.

Kuniaki Saito, Donghyun Kim, Stan Sclaroff, and Kate Saenko. Universal domain adaptation through self supervision. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Tyler Sanderson and Clayton Scott. Class proportion estimation with application to multiclass anomaly rejection. In *Artificial Intelligence and Statistics (AISTATS)*, pages 850–858, 2014.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108, 2019a.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019b.

Shibani Santurkar, Dimitris Tsipras, and Aleksander Madry. Breeds: Benchmarks for subpopulation shift. In *International Conference on Learning Representations (ICLR)*, 2021.

Nikunj Saunshi, Jordan Ash, Surbhi Goel, Dipendra Misra, Cyril Zhang, Sanjeev Arora, Sham Kakade, and Akshay Krishnamurthy. Understanding contrastive learning requires incorporating inductive biases. In *International Conference on Machine Learning*, pages 19250–19286. PMLR, 2022.

Walter J. Scheirer, Anderson de Rezende Rocha, Archana Sapkota, and Terrance E. Boult. Toward open set recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2013.

Steffen Schneider, Evgenia Rusak, Luisa Eck, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. Improving robustness against common corruptions by covariate shift adaptation. *arXiv preprint arXiv:2006.16971*, 2020.

Bernhard Schölkopf, Dominik Janzing, Jonas Peters, Eleni Sgouritsa, Kun Zhang, and Joris Mooij. On Causal and Anticausal Learning. In *International Conference on Machine Learning (ICML)*, 2012.

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems*, 35:25278–25294, 2022.

Jonathan Schwarz, Wojciech Czarnecki, Jelena Luketina, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. In *International conference on machine learning*, pages 4528–4537. PMLR, 2018.

Clayton Scott. A rate of convergence for mixture proportion estimation, with application to learning from noisy labels. In *Artificial Intelligence and Statistics*, pages 838–846, 2015.

Henry Scudder. Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory*, 11(3):363–371, 1965.

Maximilian Seitzer. pytorch-fid: FID Score for PyTorch. https://github.com/mseitzer/pytorch-fid, August 2020. Version 0.3.0.

Amrith Setlur, Don Dennis, Benjamin Eysenbach, Aditi Raghunathan, Chelsea Finn, Virginia Smith, and Sergey Levine. Bitrate-constrained dro: Beyond worst case robustness to unknown group shifts. *arXiv preprint arXiv:2302.02931*, 2023.

Harshay Shah, Kaustav Tamuly, Aditi Raghunathan, Prateek Jain, and Praneeth Netrapalli. The pitfalls of simplicity bias in neural networks. *Advances in Neural Information Processing Systems*, 33:9573–9585, 2020.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

Shai Shalev-Shwartz, Ohad Shamir, Nathan Srebro, and Karthik Sridharan. Learnability, stability and uniform convergence. *The Journal of Machine Learning Research*, 11: 2635–2670, 2010.

Kendrick Shen, Robbie M Jones, Ananya Kumar, Sang Michael Xie, Jeff Z HaoChen, Tengyu Ma, and Percy Liang. Connect, not collapse: Explaining contrastive learning for unsupervised domain adaptation. In *International Conference on Machine Learning*, pages 19847–19878. PMLR, 2022.

Jack Sherman and Winifred J Morrison. Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. *The Annals of Mathematical Statistics*, 21(1): 124–127, 1950.

Hidetoshi Shimodaira. Improving Predictive Inference Under Covariate Shift by Weighting the Log-Likelihood Function. *Journal of Statistical Planning and Inference*, 2000.

Ali Shirali and Moritz Hardt. What makes imagenet look unlike laion. *arXiv preprint arXiv:2306.15769*, 2023.

Rui Shu, Hung H Bui, Hirokazu Narui, and Stefano Ermon. A dirt-t approach to unsupervised domain adaptation. *arXiv preprint arXiv:1802.08735*, 2018.

Andrew F Siegel. Robust regression using repeated medians. *Biometrika*, 69(1):242–244, 1982.

Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in Neural Information Processing Systems*, 33, 2020.

Nimit Sohoni, Maziar Sanjabi, Nicolas Ballas, Aditya Grover, Shaoliang Nie, Hamed Firooz, and Christopher Ré. Barack: Partially supervised group robustness with guarantees. *arXiv preprint arXiv:2201.00072*, 2021.

Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro.

The implicit bias of gradient descent on separable data. *The Journal of Machine Learning Research*, 2018.

Fabio A Spanhol, Luiz S Oliveira, Caroline Petitjean, and Laurent Heutte. A dataset for breast cancer histopathological image classification. *Ieee transactions on biomedical engineering*, 63(7):1455–1462, 2015.

Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

Tejas Srinivasan, Ting-Yun Chang, Leticia Pinto Alva, Georgios Chochlakis, Mohammad Rostami, and Jesse Thomason. Climb: A continual learning benchmark for vision-and-language tasks. *Advances in Neural Information Processing Systems*, 35:29440–29453, 2022.

Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The german traffic sign recognition benchmark: a multi-class classification competition. In *International Joint Conference on Neural Networks (IJCNN)*, 2011. https://ieeexplore.ieee.org/document/6033395.

Charles M Stein. Estimation of the mean of a multivariate normal distribution. *The annals of Statistics*, pages 1135–1151, 1981.

Andreas Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your vit? data, augmentation, and regularization in vision transformers. *arXiv preprint arXiv:2106.10270*, 2021.

Gilbert W Stewart. On the early history of the singular value decomposition. *SIAM review*, 35(4):551–566, 1993.

Amos Storkey. When Training and Test Sets Are Different: Characterizing Learning Transfer. *Dataset Shift in Machine Learning*, 2009.

Arun Suggala, Adarsh Prasad, and Pradeep K Ravikumar. Connecting optimization and regularization paths. In *Advances in Neural Information Processing Systems*, pages 10608–10619, 2018.

Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *European conference on computer vision*. Springer, 2016.

Baochen Sun, Jiashi Feng, and Kate Saenko. Return of frustratingly easy domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2016.

Baochen Sun, Jiashi Feng, and Kate Saenko. Correlation alignment for unsupervised domain adaptation. In *Domain Adaptation in Computer Vision Applications*. Springer, 2017.

Richard S Sutton. Two problems with backpropagation and other steepest-descent learning procedures for networks. In *Proc. of Eightth Annual Conference of the Cognitive Science Society*, pages 823–831, 1986.

Remi Tachet, Han Zhao, Yu-Xiang Wang, and Geoff Gordon. Domain Adaptation with Conditional Distribution Matching and Generalized Label Shift. *arXiv:2003.04475 [cs, stat]*, December 2020. URL http://arxiv.org/abs/2003.04475. arXiv: 2003.04475.

Remi Tachet des Combes, Han Zhao, Yu-Xiang Wang, and Geoffrey J Gordon. Domain adaptation with conditional distribution matching and generalized label shift. *Advances in Neural Information Processing Systems*, 33, 2020.

Shuhan Tan, Jiening Jiao, and Wei-Shi Zheng. Weakly supervised open-set domain adaptation by dual-domain collaboration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5394–5403, 2019.

Shuhan Tan, Xingchao Peng, and Kate Saenko. Class-imbalanced domain adaptation: An empirical odyssey. In *European Conference on Computer Vision*, pages 585–602. Springer, 2020.

Rohan Taori, Achal Dave, Vaishaal Shankar, Nicholas Carlini, Benjamin Recht, and Ludwig Schmidt. Measuring robustness to natural distribution shifts in image classification. *Advances in Neural Information Processing Systems*, 33:18583–18599, 2020.

J. Taylor, B. Earnshaw, B. Mabey, M. Victors, and J. Yosinski. Rxrx1: An image set for cellular morphological variation across many experimental batches. In *International Conference on Learning Representations (ICLR)*, 2019.

Vishal Thengane, Salman Khan, Munawar Hayat, and Fahad Khan. Clip model is an efficient continual learner. *arXiv preprint arXiv:2210.03114*, 2022.

Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data in multimedia research. *Communications of the ACM*, 59(2):64–73, 2016.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher D Manning. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. *arXiv preprint arXiv:2305.14975*, 2023.

Yuandong Tian, Lantao Yu, Xinlei Chen, and Surya Ganguli. Understanding self-supervised learning with dual deep networks. *arXiv preprint arXiv:2010.00578*, 2020.

Ryan J Tibshirani. Adaptive piecewise polynomial estimation via trend filtering. *Annals of Statistics*, 42(1):285–323, 2014.

Antonio Torralba, Rob Fergus, and William T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1958–1970, 2008.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Joel A Tropp et al. An introduction to matrix concentration inequalities. *Foundations and Trends® in Machine Learning*, 2015.

Alexandre B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer Publishing Company, Incorporated, 1st edition, 2008.

Alexandre B Tsybakov et al. On nonparametric estimation of density level sets. *The Annals of Statistics*, 25(3):948–969, 1997.

Lifu Tu, Garima Lalwani, Spandana Gella, and He He. An empirical study on robustness to spurious correlations using pre-trained language models. *Transactions of the Association for Computational Linguistics*, 8:621–633, 2020.

Juozas Vaicenavicius, David Widmann, Carl Andersson, Fredrik Lindsten, Jacob Roll, and Thomas B Schön. Evaluating model calibration in classification. In *International Conference on Machine Learning (ICML)*, 2019.

Sara van de Geer. Estimating a regression function. *Annals of Statistics*, 18(2):907—924, 1990.

Sara van de Geer. *Empirical Processes in M-estimation*, volume 6. Cambridge university press, 2000.

Gido M Van de Ven and Andreas S Tolias. Three scenarios for continual learning. *arXiv preprint arXiv:1904.07734*, 2019.

Aad W van der Vaart and Jon A Wellner. Weak convergence. In *Weak convergence and empirical processes*. Springer, 1996.

Jesper E Van Engelen and Holger H Hoos. A survey on semi-supervised learning. *Machine learning*, 109(2):373–440, 2020.

Vladimir N Vapnik. An overview of statistical learning theory. *IEEE transactions on neural networks*, 10(5):988–999, 1999.

Bastiaan S Veeling, Jasper Linmans, Jim Winkens, Taco Cohen, and Max Welling. Rotation equivariant CNNs for digital pathology, 2018. https://arxiv.org/abs/1806.03962.

Tom Veniat, Ludovic Denoyer, and Marc'Aurelio Ranzato. Efficient continual learning with modular networks and task-driven priors. *arXiv preprint arXiv:2012.12631*, 2020.

Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5018–5027, 2017.

Roman Vershynin. *High-Dimensional Probability: An Introduction with Applications in Data Science*. Cambridge University Press, 2018.

Eli Verwimp, Kuo Yang, Sarah Parisot, Lanqing Hong, Steven McDonagh, Eduardo Pérez-Pellitero, Matthias De Lange, and Tinne Tuytelaars. Clad: A realistic continual learning benchmark for autonomous driving. *Neural Networks*, 161:659–669, 2023.

Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, pages 35151–35174. PMLR, 2023.

Tu Vu, Mohit Iyyer, Xuezhi Wang, Noah Constant, Jerry Wei, Jason Wei, Chris Tar, Yun-Hsuan Sung, Denny Zhou, Quoc Le, et al. Freshllms: Refreshing large language models with search engine augmentation. *arXiv preprint arXiv:2310.03214*, 2023.

Martin J Wainwright. *High-dimensional statistics: A non-asymptotic viewpoint*, volume 48. Cambridge university press, 2019.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=uXl3bZLkr3c.

Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019a. https://arxiv.org/abs/1905.13549.

Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. In *Advances in Neural Information Processing Systems*, pages 10506–10518, 2019b.

Jianren Wang, Xin Wang, Yue Shang-Guan, and Abhinav Gupta. Wanderlust: Online continual object detection in the real world. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10829–10838, 2021b.

Lin Wang, Hristijan Gjoreski, Mathias Ciliberto, Sami Mekki, Stefan Valentin, and Daniel Roggen. Enabling reproducible research in sensor-based transportation mode recognition with the sussex-huawei dataset. *IEEE Access*, 7:10870–10891, 2019c.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pages 9929–9939. PMLR, 2020.

Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, Mohammadhadi Bagheri, and Ronald M Summers. Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2097–2106, 2017a.

Yu-Xiang Wang, James Sharpnack, Alex Smola, and Ryan Tibshirani. Trend filtering on graphs. In *AISTATS'15*, pages 1042–1050, 2015.

Yu-Xiang Wang, James Sharpnack, Alex Smola, and Ryan J Tibshirani. Trend filtering on graphs. *Journal of Machine Learning Research*, 17(105):1–41, 2016.

Yu-Xiong Wang, Deva Ramanan, and Martial Hebert. Learning to model the tail. *Advances in neural information processing systems*, 30, 2017b.

Zifeng Wang, Zhenbang Wu, Dinesh Agarwal, and Jimeng Sun. Medclip: Contrastive learning from unpaired medical images and text. *arXiv preprint arXiv:2210.10163*, 2022a.

Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *European Conference on Computer Vision*, pages 631–648. Springer, 2022b.

Chen Wei, Kihyuk Sohn, Clayton Mellina, Alan Yuille, and Fan Yang. Crest: A class-rebalancing self-training framework for imbalanced semi-supervised learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10857–10866, 2021.

Colin Wei, Kendrick Shen, Yining Chen, and Tengyu Ma. Theoretical analysis of self-training with deep networks on unlabeled data. *arXiv preprint arXiv:2010.03622*, 2020.

Yeming Wen, Dustin Tran, and Jimmy Ba. Batchensemble: an alternative approach to efficient ensemble and lifelong learning. *arXiv preprint arXiv:2002.06715*, 2020.

Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Edouard Grave. Ccnet: Extracting high quality monolingual datasets from web crawl data. *arXiv preprint arXiv:1911.00359*, 2019.

Florian Wenzel, Andrea Dittadi, Peter Vincent Gehler, Carl-Johann Simon-Gabriel, Max Horn, Dominik Zietlow, David Kernert, Chris Russell, Thomas Brox, Bernt Schiele, et al. Assaying out-of-distribution generalization in transfer learning. *arXiv preprint arXiv:2207.09239*, 2022.

Janet Wiener and Nathan Bronson. Facebook's top open data problems. https://research.facebook.com/blog/2014/10/facebook-s-top-open-data-problems/, 10 2014. Accessed: 2023-09-28.

Olivia Wiles, Sven Gowal, Florian Stimberg, Sylvestre Alvise-Rebuffi, Ira Ktena, Taylan Cemgil, et al. A fine-grained analysis on distribution shift. *arXiv preprint arXiv:2110.11328*, 2021.

Adina Williams, Nikita Nangia, and Samuel R Bowman. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*, 2017.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45. Association for Computational Linguistics, 2020.

Ruihan Wu, Chuan Guo, Yi Su, and Kilian Q Weinberger. Online adaptation to label distribution shift. In *Advances in Neural Information Processing Systems*, 2021.

Yifan Wu, Ezra Winston, Divyansh Kaushik, and Zachary Lipton. Domain adaptation with asymmetrically-relaxed distribution alignment. In *International Conference on Machine Learning (ICML)*, 2019.

Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3733–3742, 2018.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Jianxiong Xiao, Krista A Ehinger, James Hays, Antonio Torralba, and Aude Oliva. Sun database: Exploring a large collection of scene categories. *International Journal* *of Computer Vision (IJCV)*, 2016. https://link.springer.com/article/10.1007/s11263-014-0748-y.

Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10687–10698, 2020a.

Xinpeng Xie, Jiawei Chen, Yuexiang Li, Linlin Shen, Kai Ma, and Yefeng Zheng. Self-supervised cyclegan for object-preserving image-to-image domain adaptation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*, pages 498–513. Springer, 2020b.

Da Xu, Yuting Ye, and Chuanwei Ruan. Understanding the role of importance weighting for deep learning. *arXiv preprint arXiv:2103.15209*, 2021.

Yixing Xu, Chang Xu, Chao Xu, and Dacheng Tao. Multi-positive and unlabeled learning. In *IJCAI*, pages 3182–3188, 2017.

Chhavi Yadav and Léon Bottou. Cold case: The lost mnist digits. In *Advances in Neural Information Processing Systems 32*, 2019.

Hongliang Yan, Yukang Ding, Peihua Li, Qilong Wang, Yong Xu, and Wangmeng Zuo. Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2272–2281, 2017.

Tianbao Yang, Lijun Zhang, Rong Jin, and Jinfeng Yi. Tracking slowly moving clairvoyant: optimal dynamic regret of online learning with true and noisy gradient. In *International Conference on Machine Learning (ICML-16)*, pages 449–457, 2016.

Xiangli Yang, Zixing Song, Irwin King, and Zenglin Xu. A survey on deep semi-supervised learning. *IEEE Transactions on Knowledge and Data Engineering*, 2022.

Yu Yang, Besmira Nushi, Hamid Palangi, and Baharan Mirzasoleiman. Mitigating spurious correlations in multi-modal models during fine-tuning. *arXiv preprint arXiv:2304.03916*, 2023.

Huaxiu Yao, Caroline Choi, Bochuan Cao, Yoonho Lee, Pang Wei Koh, and Chelsea Finn. Wild-time: A benchmark of in-the-wild distribution shift over time. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. URL https://openreview.net/forum?id=F9ENmZABB0.

Yuan Yao, Lorenzo Rosasco, and Andrea Caponnetto. On early stopping in gradient descent learning. *Constructive Approximation*, 26(2):289–315, 2007.

Kaichao You, Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Universal domain adaptation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2720–2729, 2019.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2014. https://aclanthology.org/Q14-1006/.

Bianca Zadrozny. Learning and Evaluating Classifiers Under Sample Selection Bias. In *International Conference on Machine Learning (ICML)*, 2004.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, pages 12310–12320. PMLR, 2021.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International conference on machine learning*, pages 3987–3995. PMLR, 2017.

Runtian Zhai, Chen Dan, Arun Suggala, J Zico Kolter, and Pradeep Ravikumar. Boosted cvar classification. *Advances in Neural Information Processing Systems*, 34:21860–21871, 2021.

Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, André Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, Lucas Beyer, Olivier Bachem, Michael Tschannen, Marcin Michalski, Olivier Bousquet, Sylvain Gelly, and Neil Houlsby. The visual task adaptation benchmark, 2019. http://arxiv.org/abs/1910.04867.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations (ICLR)*, 2017.

Dell Zhang and Wee Sun Lee. A simple probabilistic approach to learning from positive and unlabeled examples. In *Proceedings of the 5th Annual UK Workshop on Computational Intelligence (UKCI)*, pages 83–87, 2005.

Hongjie Zhang, Ang Li, Jie Guo, and Yanwen Guo. Hybrid models for open set recognition. In *European Conference on Computer Vision*, pages 102–117. Springer, 2020.

Jingzhao Zhang, Aditya Menon, Andreas Veit, Srinadh Bhojanapalli, Sanjiv Kumar, and Suvrit Sra. Coping with label shift via distributionally robust optimisation. In *International Conference on Learning Representations (ICLR)*, 2021.

Kun Zhang, Bernhard Schölkopf, Krikamol Muandet, and Zhikun Wang. Domain Adaptation Under Target and Conditional Shift. In *International Conference on Machine Learning (ICML)*, 2013.

Lijun Zhang, Shiyin Lu, and Zhi-Hua Zhou. Adaptive online learning in dynamic environments. In *Advances in Neural Information Processing Systems (NeurIPS-18)*, pages 1323–1333, 2018a.

Lijun Zhang, Tianbao Yang, Zhi-Hua Zhou, et al. Dynamic regret of strongly adaptive methods. In *International Conference on Machine Learning (ICML-18)*, pages 5877–5886, 2018b.

Qi Zhang, Yifei Wang, and Yisen Wang. On the generalization of multi-modal contrastive learning. *arXiv preprint arXiv:2306.04272*, 2023.

Richard Zhang. Making convolutional networks shift-invariant again. In *ICML*, 2019.

Weichen Zhang, Wanli Ouyang, Wen Li, and Dong Xu. Collaborative and adversarial network for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018c.

Yuchen Zhang, Tianle Liu, Mingsheng Long, and Michael Jordan. Bridging theory and algorithm for domain adaptation. In *International Conference on Machine Learning*. PMLR, 2019.

Yuhao Zhang, Hang Jiang, Yasuhide Miura, Christopher D Manning, and Curtis P Langlotz. Contrastive learning of medical visual representations from paired images and text. In *Machine Learning for Healthcare Conference*, pages 2–25. PMLR, 2022.

Han Zhao, Remi Tachet Des Combes, Kun Zhang, and Geoffrey Gordon. On learning invariant representations for domain adaptation. In *International Conference on Machine Learning*, pages 7523–7532. PMLR, 2019.

Peng Zhao and Lijun Zhang. Improved analysis for dynamic regret of strongly convex and smooth functions. *L4DC*, 2021.

Peng Zhao, Y. Zhang, L. Zhang, and Zhi-Hua Zhou. Dynamic regret of convex and smooth functions. *NeurIPS*, 2020.

Peng Zhao, Yu-Xiang Wang, and Zhi-Hua Zhou. Non-stationary online learning with memory and non-stochastic control. *AISTATS*, 2022.

Zangwei Zheng, Mingyuan Ma, Kai Wang, Ziheng Qin, Xiangyu Yue, and Yang You. Preventing zero-shot transfer degradation in continual learning of vision-language models. *arXiv preprint arXiv:2303.06628*, 2023.

Da-Wei Zhou, Fu-Yun Wang, Han-Jia Ye, and De-Chuan Zhan. Pycil: A python toolbox for class-incremental learning, 2023a.

Da-Wei Zhou, Yuanhan Zhang, Jingyi Ning, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Learning without forgetting for vision-language models. *arXiv preprint arXiv:2305.19270*, 2023b.

Lijia Zhou, Danica J Sutherland, and Nathan Srebro. On uniform convergence and low-norm interpolation learning. *arXiv preprint arXiv:2006.05942*, 2020.

Wenda Zhou, Victor Veitch, Morgane Austern, Ryan P Adams, and Peter Orbanz. Non-vacuous generalization bounds at the imagenet scale: a pac-bayesian compression approach. *arXiv preprint arXiv:1804.05862*, 2018.

Xiaojin Zhu and Zoubin Ghahramani. Learning from labeled and unlabeled data with label propagation. *CMU CALD tech report CMU-CALD-02-107, 2002*, 07 2003.

Yang Zou, Zhiding Yu, BVK Kumar, and Jinsong Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *Proceedings of the European conference on computer vision (ECCV)*, pages 289–305, 2018.