Collaborative learning by leveraging siloed data

Sebastian Caldas

July 2023
CMU-ML-23-106

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

Thesis Committee:
Artur Dubrawski, Chair
Virginia Smith
Gilles Clermont (University of Pittsburgh)
Martin Jaggi (EPFL)

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

This research was sponsored by: Air Force Research Laboratory award FA87501720130; Defense Advanced Research Projects Agency award HR00111830004; National Institutes of Health awards R01GM117622, R01NR013912, R01HL144692, R01HL141916, R01EB029751 and R01NR013912; National Science Foundation awards IIS1705121 and IIS1838017; United States Army awards W911NF1820218 and W81XWH19C0101.

Keywords: machine learning, collaborative learning, federated learning, machine learning for healthcare, model compression, interpretability, weak supervision.

Para mi madre

Abstract

Regulations can often limit stakeholders' modeling capabilities by preventing data sharing. For example, in order to protect patient privacy, clinical centers may be unable to share their data and thus lack representative records to learn about a rare condition. To address this challenge, previous work in machine learning has shown that these stakeholders benefit from training models in a collaborative fashion, improving their predictive performance. However, as we start training these collaborative models in real-world settings, and in order to be truly useful, they need to provide utility along dimensions beyond predictive performance. In this thesis, we propose methods and algorithms to improve collaborative models that leverage siloed data along three dimensions. In the first part, we propose methods to reduce the communication footprint of models learned by mobile devices cooperating over edge networks, allowing for higher capacity models to be trained. Then, in the second part, we introduce an algorithm that provides explanations about predictions of models trained across clinical centers, thus improving their clinical utility. Finally, in the third part, we address the need to encode expert supervision into collaborative models trained using on-device data, increasing the class of problems we can tackle in these scenarios.

Acknowledgements

I want to thank my advisor, Artur: for answering every question with the utmost patience, for giving me space to explore ideas at my own pace, and for teaching me what great mentorship can look like. I strive to create spaces as impactful as your lab. I also want to thank Ginger, for her compassionate guidance, support and example, especially in the early stages of my PhD. Finally, I would also like to thank the rest of my committee, Dr. Clermont and Professor Jaggi, for their invaluable advice. It has had a profound impact on my work.

I am very grateful to have grown as a researcher alongside extraordinary collaborators in academia, industry and healthcare. The list includes but is probably not limited to Jakub Konečnỳ, Brendan McMahan, Michael Pinsky, Joo Yoon, Keith Dufendach, Jieshi Chen, Vivek Natarajan, Yuan Liu, Christopher Choquette, Mikhail Khodak, Tian Li, Mononito Goswami, Jeffrey Li, Karthik Duddu, Peter Wu, Vincent Jeanselme, Willa Potosnak, and Kyle Miller.

I am thankful to Ameet Talwalkar and Rayid Ghani, for allowing me to teach in their classrooms. At the time, I did not articulate how much that opportunity meant to me. I also want to highlight the kindness and passion of Diane Stidle. I thank my undergraduate mentors, Marcela Hernandez and Rubby Casallas, for going out of their way to let me pursue machine learning research.

I am incredibly lucky to have landed in the AutonLab, and I am thankful for the friends I found there: Ceci, Ifi, Cristian, Ian, Angela, Jack, Roman, Rachel, Youngseog, Ben, Chirag, Aleeya and Nick. I am also thankful for the friends I have found in CMU: Nicholay, Mandy, Devendra, Jin, Jen, Ken, Kanthashree, Sid, Biswa, Shaojie, Kin, Stef, Eyan, Liam, Greg, Giulio, Jason, Lisa, Conor and David. Thanks to Mariya Toneva, because her advice kept me going during tumultuous times. Finally, I'm grateful to my office-mates: Leqi, Brandon and Helen. Thank you all for sharing your joy and your depth and your company.

I also want to acknowledge the support of my Pittsburgh friends. The improv teams at SCIT always had my back: All in One Peel, That New Jersey Smell, Updates Pending, The Unnamed Harold Team, among many others. I'm thankful to the friends that had my back beyond improv: Miguelina, Emma, Oisin, Maia, Nick, Cam, Kelly, Quinn, Ray, Robin, Patrick and Nat. You made me feel at home in Pittsburgh, something I struggled to do for a while.

Para finalizar, quiero agradecer a mi familia. A mi hermana, por su apoyo y su consejo. A mi madre y a mi padre, por cultivar mi amor por el aprendizaje. Y a mi abuela y a mis tías, por su constante afecto.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem Statement

Data is essential for modern machine learning systems (Gebru et al., 2021). However, misaligned incentives and privacy regulations often prevent data holders from sharing their data (Van Panhuis et al., 2014; Rieke et al., 2020), limiting our modeling capabilities in scenarios such as healthcare and mobile computing. For example, a hospital may lack representative records to learn about a new or rare condition (Wiens et al., 2014), or a single mobile device may not have enough input to train a useful language model about its user. In both of these cases, each siloed data holder would benefit from collaborating with others in order to leverage their data.

In recent years, the machine learning community has taken an interest in learning collaborative models from siloed data. Previous work has shown these models to be performant in domains such as healthcare (Andreux et al., 2020a; Caldas et al., 2020; Sadilek et al., 2021), finance (Zheng et al., 2020) and mobile computing (Hard et al., 2018; Ramaswamy et al., 2019). However, due to their nature as a collaboration, these models are part of systems that provide utility along axes beyond predictive performance, such as confidentiality (Bonawitz et al., 2017), fairness (Li et al., 2020c) and privacy (McMahan et al., 2018; Li et al., 2019a).

In this thesis, we are interested in the axes of utility that allow collaborative learning to tackle tasks of real-world importance (see Table 1.1). This implies satisfying the practical constraints of these settings, e.g., adhering to a communication budget over a mobile network. Furthermore, this requires understanding and satisfying users' requirements over the collaborative learning systems. For example, for clinicians to regularly use a model, they may require both clinical-grade performance and explanations for the given predictions.

This dissertation addresses methodological and algorithmic challenges of learning in settings where there are multiple data holders who wish to leverage each other's data. The base constraint is that each holder cannot share its own data. However, in each part of this thesis we confront and address additional practical constraints with the objective of making these models truly practical and useful.

Dimension of utility	Related Work
Communication constraints (Part I)	Konečný et al. (2016); Caldas et al. (2019b); Vogels et al. (2019); Horvath et al. (2021)
Explanations (Part II)	Caldas et al. (2021b,a); Potosnak et al. (2021); Bárcena et al. (2022)
Expert Supervision (Part III)	Jeong et al. (2020); Wu et al. (2021); Liu et al. (2021); Li et al. (2022)
Privacy	McMahan et al. (2018); Li et al. (2019a); Adnan et al. (2022); Bonawitz et al. (2022)
Personalization	Smith et al. (2017); Khodak et al. (2019a); Mansour et al. (2020); Paulik et al. (2021)
Fairness	Mohri et al. (2019); Li et al. (2020c); Haghtalab et al. (2022)

Table 1.1: Examples of dimensions of utility studied in the context of collaborative learning systems that leverage siloed data. For each dimension, we provide a limited list of related work.

Collaborative Learning by Leveraging Siloed Data

In *collaborative learning*, k stakeholders with distributions $D_1, \ldots, D_k$ are labeled according to an unknown function $f^* \in \mathcal{F}$. The goal is then to collaboratively learn a model to approximate f^* up to an error ϵ on each stakeholder's distribution (Blum et al., 2017; Haghtalab et al., 2022).

Blum et al. (2017) show that collaborative learning is more sample efficient than naive algorithms that do not exchange information among stakeholders. This follows the vein of other frameworks that exploit structure across multiple tasks in order to decrease sample complexity and facilitate learning, e.g., multi-task learning (Ben-David & Schuller, 2003), multi-source domain adaptation (Mansour et al., 2008, 2009) and hypothesis transfer learning (Kuzborskij & Orabona, 2013). Data siloing, however, is not an explicit restriction in Blum et al. (2017).

In this thesis, we will formalize the notions of collaboration and siloing using the framework provided by *federated learning*: a collaborative learning setting that keeps the data siloed (McMahan et al., 2017; Caldas et al., 2019a). This siloing allows federated learning to follow the principles of focused data collection and minimization, opting instead to communicate focused updates intended for aggregation (Kairouz et al., 2021; Bonawitz et al., 2022).

Using this framework, we will distinguish between two federated learning scenarios that will contextualize the work in this thesis (Kairouz et al., 2021):

- **Cross-device:** Refers to scenarios such as mobile computing in which the collaborators have limited learning resources, e.g., compute and data. The small per-collabotor resources are offset by a massive number or collaborators.
- **Cross-silo:** In these setting, the per-collaborator learning resources are not an issue. This setting is usually characterized by a small number of collaborators (< 100).

1.2 Thesis Statement and Overview

This thesis focuses on improving the practical utility of collaborative learning systems that leverage siloed data. With this in mind, this dissertation is centered around the following thesis statement:

> *Practical utility in collaborative learning systems requires attributes beyond predictive performance. In particular, when leveraging siloed data, practical utility improves when including attributes such as a reduced communication footprint, explanations, and the ability to encode expert supervision.*

To substantiate this idea, we organize this thesis into three main parts:

Part I: Communication Constraints

In this part, we support our claim by studying a collaborative learning setting where communication is a fundamental bottleneck: cross-device federated learning over heterogeneous edge networks (Caldas et al., 2019b). We argue how this bottleneck restricts both the capacity of the model being trained, and the collaborators who can actively participate in the training procedure. We then propose two novel strategies to reduce communication costs based on lossy compression and federated dropout. Lastly, we empirically show that these strategies provide a reduction in communication without degrading the accuracy of the final model.

Part II: Explanations

In this part, we substantiate our central ideal by studying cross-silo clinical collaborations, where stakeholders require both explanations and predictive power to derive true clinical utility (Caldas et al., 2021b). We introduce FRCLS[1], an algorithm that explicitly identifies when a new prediction is using knowledge from an external collaborator, and provides interpretable rules delineating subpopulations for which that external knowledge is useful. We evaluate this algorithm on a variety of clinical tasks (Potosnak et al., 2021) and propose a connection with the literature on transferring best practices across clinical centers (Caldas et al., 2021a).

Part III: Expert Supervision

This part provides support for our thesis statement by highlighting how expert annotations are often needed yet unavailable in cross-device collaborative settings. We tackle this challenge by proposing WSHFL[2], an algorithms that uses feedback from an expert situated at the server to learn heuristics which can label the devices' data (Caldas et al., 2023). Later on, WSHFL trains a weak supervision model using those heuristics. We validate our approach on datasets across data modalities, showing we are competitive against a fully supervised baseline.

[1] pronounced as in freckles.
[2] pronounced as in wishful.

1.3 Summary of Contributions

We summarize the contributions in this thesis:

- Part I: We address the open question as to whether lossy compression approaches are amenable for server-to-client exchanges in the context of cross-device federated learning.
- Part I: We introduce Federated Dropout, a technique that enables each collaborator to locally operate on a small sub-section of a larger global model being trained. This technique reduces communication costs by allowing for these sub-models to be exchanged.
- Part I: We empirically show that lossy compression and Federated Dropout are compatible with one another, reducing the communication footprint without degrading the model's accuracy.
- Part II: We argue that clinical utility for collaborative systems requires explaining how the collaboration itself is affecting a center's predictions, i.e., whether a decision is being made based on knowledge from an external center.
- Part II: We introduce an algorithm that produces rules that clearly delineate the regions of the feature space where external centers are more competent than the local one, providing interpretable rationale for decisions made by local stakeholders.
- Part II: We test our proposed approach on a benchmark sepsis prediction task in two hospital systems, showing that it is capable of providing both a boost in predictive power and interpretable insights into the types of patients most benefited by the collaboration. Further extensions also show the benefits of our method on tasks such as prediction of hypotension and early prediction of mortality across ICUs.
- Part II: We use our approach to propose a methodology that identifies apparent practice gaps between clinical centers, and provides a stop-gap solution while the practice transfer takes place. We demonstrate the potential of our methodology in the context of predicting overly long lengths of stay.
- Part III: We introduce Programmatic Weak Supervision (PWS) into the cross-device federated setting, encoding domain knowledge into on-device data by interactively querying an expert to inspect candidate Labeling Functions (LFs) mined from the data.
- Part III: We propose approaches for two components of a standard PWS workflow in a federated set-up: the generation of candidate LFs, and the training of a PWS model given LFs selected by the expert.
- Part III: We validate the feasibility of our approach on on three datasets across two data modalities, demonstrating competitive performance compared to a fully supervised baseline while reducing the need for direct data annotations. In particular, we are amongst the first to learn classification models from unlabeled distributed time-series data.

1.4 Bibliographic Notes

Most of the work in this thesis has been published or has been presented at the following venues:

Part I, Chapter 2 is partially based on:

- Sebastian Caldas, Jakub Konečnỳ, H Brendan McMahan, and Ameet Talwalkar. Expanding the reach of federated learning by reducing client resource requirements. *Workshop on Federated Learning for Data Privacy and Confidentiality at NeurIPS*, 2019b. URL https://arxiv.org/abs/1812.07210.

Part II, Chapter 3 is partially based on:

- Sebastian Caldas, Joo Heung Yoon, Michael R. Pinsky, Gilles Clermont, and Artur Dubrawski. Understanding clinical collaborations through federated classifier selection. In Ken Jung, Serena Yeung, Mark Sendak, Michael Sjoding, and Rajesh Ranganath (eds.), *Proceedings of the 6th Machine Learning for Healthcare Conference*, volume 149 of *Proceedings of Machine Learning Research*, pp. 126–145. PMLR, 06–07 Aug 2021b. URL https://proceedings.mlr.press/v149/caldas21a.html.

Part II, Chapter 4 is partially based on:

- Sebastian Caldas, Jieshi Chen, and Artur Dubrawski. Using machine learning to support transfer of best practices in healthcare. In *AMIA Annual Symposium Proceedings*, volume 2021, pp. 265. American Medical Informatics Association, 2021a. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8861698/.

Part III, Chapter 5 is partially based on:

- Sebastian Caldas, Mononito Goswami, and Artur Dubrawski. Encoding expert knowledge into federated learning using weak supervision. *ICLR 2023 workshop on Machine Learning for IoT: Datasets, Perception, and Understanding*, 2023.

Since Parts I to III concern previously presented results, the work on this thesis unifies their discussion, and presents new research ideas. In particular, we expand on the work that we have previously presented for Part III.

1.5 Open-Source Contributions

We argue that the developments made in cross-device federated learning should be grounded with realistic benchmarks. In Part IV, we take a step in this direction by introducing LEAF, a benchmarking framework for learning with on-device data.

Part IV, Chapter 6 is thus partially based on:

- Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečnỳ, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. LEAF: A benchmark for federated settings. *Workshop on Federated Learning for Data Privacy and Confidentiality at NeurIPS*, 2019a. URL https://arxiv.org/abs/1812.01097.

Part I

Communication Constraints

Communication on heterogeneous edge networks is a fundamental bottleneck when learning collaborative models from on-device data, restricting both model capacity and user participation. To address this issue, we introduce two novel strategies to reduce communication costs when learning in these scenarios: (1) the use of lossy compression on models downloaded by the users; and (2) Federated Dropout, which allows users to exchange smaller subsets of the global model. We empirically show that these strategies, combined with existing compression approaches for the gradients uploaded by the users, collectively provide a reduction in the communication and computation footprint of collaborative models, all without degrading the quality of the final model. Thus, we thus comprehensively reduce the impact of collaborative training on client device resources, allowing higher capacity models to be trained, and a more diverse set of users to be reached.

Chapter 2

Expanding the Reach of Federated Learning by Reducing Client Resource Requirements

We aim to reap the benefits of collaborative models trained from the rich yet sensitive data captured by mobile devices, without the need to centrally store such data (McMahan et al., 2017). To learn these models, we leverage the cross-device Federated Learning (FL) paradigm, in which devices collaborate by performing training on samples available locally and only communicate intermediate model updates with a central server (Kairouz et al., 2021; Wang et al., 2021).

Network speed and number of nodes are two of the core systems aspects that differentiate cross-device FL from traditional distributed learning in data centers, with network bandwidth being potentially orders of magnitude slower and the number of worker nodes orders of magnitude larger. Together, these issues exacerbate the communication bottlenecks usually associated with distributed learning, increasing both the number of stragglers and the probability of devices dropping out altogether.

This communication bottlenecks are further aggravated when working with high capacity models with large numbers of parameters. Insisting on training these large models using naive federated optimization methods can lead to the systematic exclusion of clients with restricted bandwidth or limited network access, which tend to be older users in rural areas (Anzilotti, 2016; Pew Research Center, 2018). Naive training will thus lead to a degraded user experience once the resulting models are served to these populations.

One possible solution involves training low capacity collaborative models with smaller communication footprints, but this comes at the expense of model accuracy. As a middle ground, we could develop strategies to reduce the communication footprint of larger, high-capacity models. Recent work (Konečný et al., 2016) has in fact taken this approach, but only in the context of the information exchanged client-to-server. Their success with lossy compression strategies is perhaps not surprising, as the clients' lossy, yet unbiased, updates are eventually averaged over many users. However, server-to-client exchanges do not benefit from such averaging. As such, they remain a bottleneck in our goal of truly reaping the benefits of collaborative models trained from on-device data.

In this chapter, we propose two novel strategies to mitigate the server-to-client communication footprint, and empirically demonstrate their efficacy and seamless integration with

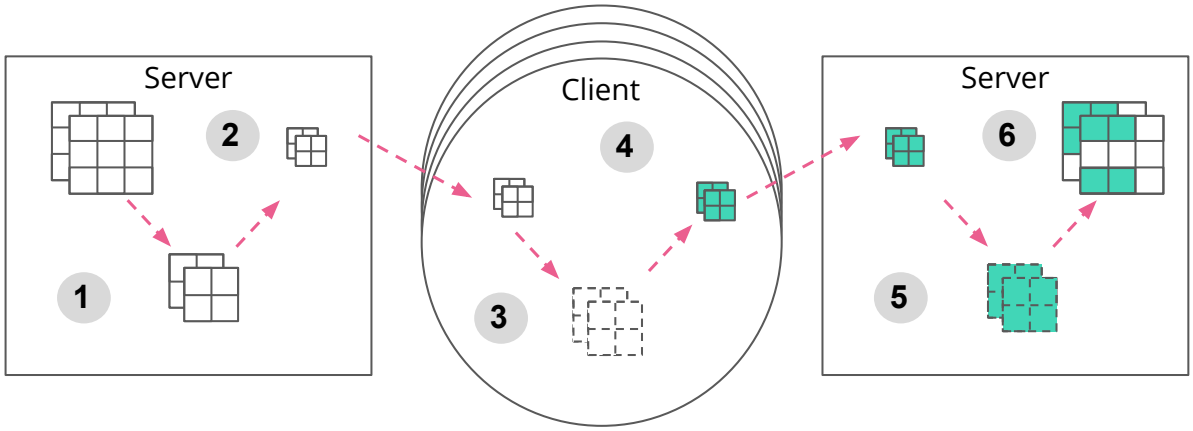

Figure 2.1: Combination of our proposed strategies during FL training. We reduce the size of the model to be communicated by (1) constructing a sub-model via *Federated Dropout*, and by (2) lossily compressing the resulting object. This compressed model is then sent to the client, who (3) decompresses and trains it using local data, and (4) compresses the final update. This update is sent back to the server, where it is (5) decompressed and finally, (6) aggregated into the global model.

existing client-to-server strategies. The specific contributions of this chapter are as follows:

1. We study lossily compressing the models downloaded by the clients, thus addressing the open question as to whether these approaches are amenable in the context of server-to-client exchanges. We also introduce the use of the theoretically motivated Kashin's representation to reduce the error associated with the lossy compression (Lyubarskii & Vershynin, 2010; Kashin, 1977).
2. We introduce *Federated Dropout*, a technique that builds upon the popular idea of dropout (Srivastava et al., 2014), yet is primarily motivated by systems-related concerns. Our approach enables each device to locally operate on a *smaller* sub-model (i.e. with smaller weight matrices) while still providing updates that can be applied to the larger global model on the server. It thus reduces communication costs by allowing for these smaller sub-models to be exchanged between server and clients, while also reducing the computational cost of local training.
3. We empirically show that not only are these approaches compatible with one another, but with existing client-to-server compression. Combining these approaches during FL training (see Figure 2.1) reduces the size of the downloaded models, the size of the corresponding updates, and the required local computations, all without degrading the model's accuracy and only at the expense of a slightly slower convergence rate (in terms of number of communication rounds).

2.1 Related Work

Federated Learning. FL is a technique that aims to learn a global model over data distributed across multiple edge devices (usually mobile phones) without the data ever leaving the device on which it was generated (McMahan et al., 2017; Kairouz et al., 2021). It brings along a set of statistical (non-IID, unbalanced data) and systems (stragglers, communication bottlenecks, etc.) challenges which differentiate it from traditional distributed learning in the data center.

In this work we study Federated Averaging (FedAvg), an algorithm proposed by McMahan et al. (2017). In its canonical form, this algorithm works by (1) sending the global model to a subset of the available devices, (2) training the model on each device using the available local data, and (3) averaging the local updates to thus end a round of training. FedAvg is part of a family of local SGD methods (Wang & Joshi, 2021) that learn collaborative models from on-device data (Reddi et al., 2020) by continually exchanging information across a potentially slow network.

Communication-efficient distributed learning. Distributed learning is known to suffer from communication overheads associated with the frequent gradient updates exchanged among nodes (Wang et al., 2018; Dean et al., 2012; Smith et al., 2018; Reddi et al., 2016). To reduce these bottlenecks, previous studies have communicated a sparsified (Stich et al., 2018; Alistarh et al., 2018) or quantized (Alistarh et al., 2017; Suresh et al., 2017a; Wu et al., 2018) version of the updates. Although these operations introduce noise, they have been shown both empirically and theoretically to maintain the quality of the trained models.

In the context of FL, Konečný et al. (2016); Haddadpour et al. (2021); Wang et al. (2022) study lossy compression on the client-to-server exchanges (i.e. the model updates). Of particular interest is the use of the randomized Hadamard transform by Konečný et al. (2016) to reduce the error incurred by the subsequent quantization, as it spreads a vector's information more evenly across its components (Suresh et al., 2017b; Konečný & Richtárik, 2016).

Previous work in FL does not consider compressing the server-to-client exchanges. Nevertheless, downloading a large model can still be a considerable burden for users, particularly for those in regions with network constraints. Furthermore, as FL is expected to deal with a large number of devices, communicating the global model may even become a bottleneck for the server, as it would send the model to the clients in parallel. Previous work in distributed learning that researches this bi-directional compression includes Tang et al. (2019); Zheng et al. (2019); Horvóth et al. (2022).

Model compression. Deep models tend to demand significant computational resources for training and inference. Using them on edge devices is thus not a straightforward task. Hence, several recent works have proposed compressing the models before deploying them on-device, e.g., via pruning (Han et al., 2016; Fang et al., 2023), weight quantization (De Sa et al., 2018; Xiao et al., 2023), and distillation (Hinton et al., 2015; Dennis et al., 2023). Many of these approaches, however, are not applicable for the problems addressed in this chapter, as they are either ingrained in the training procedure or are mostly optimized for inference. In the context of FL, we need an algorithm that is computationally light, can be efficiently applied in every round, and allows for subsequent local training. We do note, however, that some of the previously mentioned approaches can be leveraged at inference time in the federated setting.

2.2 Methods

In this section, we present our proposed strategies for reducing FL's server-to-client communication costs, namely lossy compression techniques (Section 2.2.1) and *Federated Dropout* (Section 2.2.2). We introduce the strategies separately, but they are fully compatible with one another (as we show in Section 2.3.4).

2.2.1 Lossy Compression

Our first approach at reducing bandwidth usage consists of using lightweight lossy compression techniques that can be applied to an already trained model and that, when reversed (i.e. after decompression), maintain the model's quality. The particular set of techniques we propose are inspired by those successfully used by Konečný et al. (2016) to compress the client-to-server updates. We apply them, however, to the server-to-client exchanges, meaning we do not get the benefit of averaging the noisy decompressions over many updates.

Our method works as follows: we reshape each to-be-compressed weight matrix in our model into a vector $\boldsymbol{w}$ and (1) apply a basis transform to it. We then (2) subsample and (3) quantize the resulting vector and finally send it through the network. Once received, we simply execute the respective inverse transformations to finally obtain a noisy version of $\boldsymbol{w}$.

Basis transform. Previous work (Lyubarskii & Vershynin, 2010; Konečný et al., 2016) has explored the idea of using a basis transform to reduce the error that will later be incurred by perturbations such as quantization. In particular, Konečný et al. (2016) use the random Hadamard transform to more evenly spread out a vector's information among its dimensions. We go even further and also apply the classical results of Kashin (1977) to spread a vector's information *as much as possible* in every dimension (Lyubarskii & Vershynin, 2010). Thus, Kashin's representation mitigates the error incurred by subsequent quantization compared to using the random Hadamard transform. For a more detailed discussion, we refer the reader to Section A.1.3 in the Appendix.

Subsampling. For $s \in [0, 1)$, we zero out a $1 - s$ fraction of the elements in each weight matrix, appropriately re-scaling the remaining values. The elements to zero out are picked uniformly at random. Thus, we only communicate the non-zero values and a random seed which allows recovery of the corresponding indices.

Probabilistic quantization. For a vector $\boldsymbol{w} = (w_1, \ldots, w_n)$, let us denote $w_{\min} = \min_j \{w_j\}_{j=1}^n$ and $w_{\max} = \max_j \{w_j\}_{j=1}^n$. Uniform probabilistic 1-bit quantization replaces every element w_i by $w_{\min}$ with probability $\frac{w_{\max} - w_i}{w_{\max} - w_{\min}}$, and by $w_{\max}$ otherwise. It is straightforward to verify this yields an unbiased estimate of $\boldsymbol{w}$. Now, for q-bit uniform quantization, we first equally divide $[w_{\min}, w_{\max}]$ into 2^q intervals. If w_i falls in the interval bounded by w' and w'', the quantization operates by replacing $w_{\min}$ and $w_{\max}$ in step two of the above algorithm by w' and w'', respectively.

2.2.2 Federated Dropout

To further reduce communication costs, we propose an algorithm in which each client, instead of locally training an update to the whole global model, trains an update to a smaller sub-model.

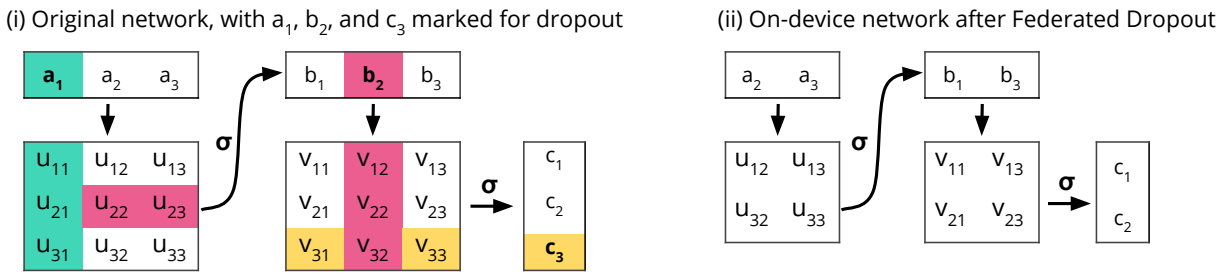

Figure 2.2: *Federated Dropout* applied to two fully-connected layers. Notices activation vectors $a, b = \sigma(Ua)$ and $c = \sigma(Vb)$ in (I). In this example, we randomly select exactly one activation from each layer to drop, namely a_1, b_2, and c_3, producing a sub-model with 2×2 dense matrices, as in (II).

These sub-models are subsets of the global model and, as such, the computed local updates have a natural interpretation as updates to the larger global model. We call this technique *Federated Dropout* as it is inspired by the well known idea of dropout (Srivastava et al., 2014), albeit motivated primarily by systems-level concerns rather than as a strategy for regularization.

In traditional dropout, hidden units are multiplied by a random binary mask in order to drop an expected fraction of neurons during each training pass through the network. Because the mask changes in each pass, each pass is effectively computing a gradient with respect to a different sub-model. These sub-models can have different sizes (architectures) depending on how many neurons are dropped in each layer. Now, even though some units are dropped, in all implementations we are aware of, activations are still multiplied with the original weight matrices, they just have some useless rows and columns.

To extend this idea to FL and realize communication and computation savings, we instead zero out a *fixed* number of activations at each fully-connected layer, so all possible sub-models have the same reduced architecture; see Figure 2.2. The server can map the necessary values into this reduced architecture, meaning only the necessary coefficients are transmitted to the client, re-packed as smaller dense matrices. The client (which may be fully unaware of the original model's architecture) trains its sub-model and sends its update, which the server then maps back to the global model[1]. For convolutional layers, zeroing out activations would not realize any space savings, so we instead drop out a fixed percentage of filters.

This technique brings two additional benefits beyond savings in server-to-client communication. First, the size of the client-to-server updates is also reduced. Second, the local training procedure now requires a smaller number of FLOPS per gradient evaluation, either because all matrix-multiplies are now of smaller dimensions (for fully-connected layers) or because less filters have to be applied (for convolutional ones). Thus, we reduce local computational costs.

[1]This can be done by communicating a single random seed to the client and back, or via state on the server.

Dataset	# of users	IID	Training samples per user		Test samples per user	
			mean	σ	mean	σ
MNIST	100	Yes	600	0	100	0
CIFAR-10	100	Yes	500	0	100	0
EMNIST	3550	No	181.46	71.15	45.37	17.79

Table 2.1: Summary of datasets used in the experiments.

2.3 Experimental Results

In this section, we first present our experimental setup (Section 2.3.1) before presenting results for our lossy compression (Section 2.3.2) and *Federated Dropout* (Section 2.3.3) strategies. Finally, we show experiments that use both of these strategies in tandem with those proposed in Konečný et al. (2016) to also compress client-to-server exchanges (Section 2.3.4).

2.3.1 Experimental Setup

Optimization Algorithm. We focus on testing our strategies against already established FL benchmarks. In particular, we restrict our experiments to the use of Federated Averaging (FedAvg) (McMahan et al., 2017).

Datasets. We use three datasets in our experiments: MNIST (LeCun et al., 1998), CIFAR-10 (Krizhevsky & Hinton, 2009) and Extended MNIST or EMNIST (Cohen et al., 2017). The first two were used to benchmark the performance of FedAvg and of lossy compression for client-to-server updates (Konečný et al., 2016). For these two datasets, we use the artificial IID partition proposed by these previous works. Meanwhile, EMNIST is a dataset that has only recently been introduced as a useful benchmark for FL. Derived from the same source as MNIST, it also includes the identifier of the user that wrote the character (digit, lower or upper case letter), creating a natural and much more realistic partition of the data. Table 2.3.1 summarizes the basic dataset properties. Due to space constraints, we relegate the MNIST results to Appendix A.2, though all conclusions presented here also qualitatively hold for these experiments.

Models. For MNIST's digit recognition task we use the same model as McMahan et al. (2017): a CNN with two 5x5 convolution layers (the first with 32 channels, the second with 64, each followed by 2x2 max pooling), a fully connected layer with 512 units and ReLu activation, and a final softmax output layer, for a total of more than 10^6 parameters. For CIFAR-10, we use the all convolutional model taken from what is described as "Model C" in Springenberg et al. (2015), which also has a total of over 10^6 parameters. Finally, for EMNIST we use a variant of the MNIST model with 2048 units in the final fully connected layer. While none of these models is the state-of-the-art, they are sufficient for evaluating our methods, as we wish to measure accuracy degradation against a baseline and not to achieve the best possible accuracy on these tasks.

Hyperparameters. We do not optimize our experiments for FedAvg's hyperparameters, always using those that proved to work reasonably well in our baseline setting which involves

no compression and no *Federated Dropout*. For local training at each client we use static learning rates of 0.15 for MNIST, 0.05 for CIFAR-10 and 0.035 for EMNIST. We select 10 random clients per round for MNIST and CIFAR-10, and 35 for EMNIST. Finally, each selected client trains for one epoch per round using a batch size of 10.

2.3.2 Lossy Compression

We focus on testing how the compression strategies presented in Section 2.2.1 impact the global model's accuracy. Like Konečný et al. (2016), we don't compress all variables of our models. As they mention, compressing smaller variables causes significant accuracy degradation but translates into minuscule communication savings. As such, we don't compress biases for any of the models[2].

In our experiments, we vary three parameters:

1. The type of basis transform applied: no transform or identity (I), randomized Hadamard transform (HD) and Kashin's representation (K).
2. The subsampling rate s, which refers to the fraction of weights that are kept (i.e. $1 - s$ of the weights are zeroed out).
3. The number of quantization bits q.

Figure 2.3 shows the effect of varying these parameters for CIFAR-10 and EMNIST. We repeat each experiment 10 times and report the mean accuracy among these repetitions. The three main takeaways from these experiments are: (1) for every model, we are able find a setting of compression parameters that at the very least matches our baseline; (2) Kashin's representation proves to be most useful for aggressive quantization values; and (3) it appears that subsampling is not all that helpful in the server-to-client setting. We proceed to give more details about these highlights.

The first takeaway is that, for every model, we are indeed able find a setting of compression parameters that matches or, in some cases, slightly outperforms our baseline. In particular, we are able to quantize every model to 4 bits, which translates to a reduction in communication of nearly $8\times$.

The second takeaway is that Kashin's representation proves to be most useful for aggressive quantization values, i.e. for low values of q. In our experiments, gains were observed only in regimes where the overall accuracy had already degraded, but we hypothesize that the use of Kashin's representation may provide clearer benefits in the compression of client-to-server gradient updates, where more aggressive quantization is admissible. We also highlight that using Kashin's representation may be beneficial for other datasets. Indeed, its computational costs are comparable to that of the random Hadamard transform while also providing better theoretical error rates (see Section A.1.1). We refer the reader to Section A.1.3 in the Appendix, where we show preliminary results that demonstrate Kashin's potential to dominate over the randomized Hadamard transform in compressing fully-trained models, particularly for small values of q.

[2]Unlike Konečný et al. (2016), we do compress all 9 convolutional layers in the CIFAR-10 model, not just the 7 in the middle.

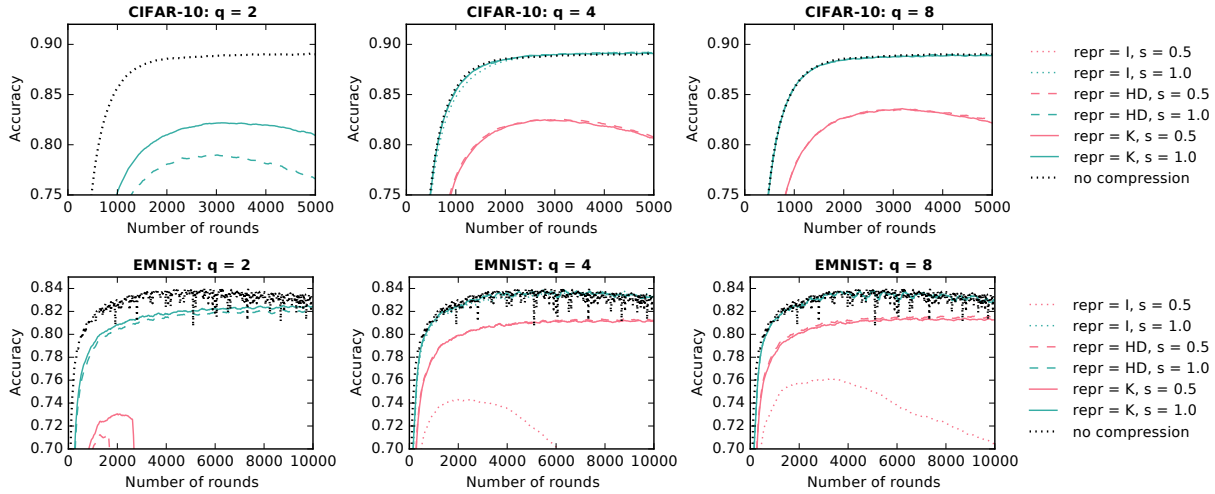

Figure 2.3: Effect of varying our lossy compression parameters on CIFAR-10 and EMNIST.

Finally, it appears that subsampling is not all that helpful in this server-to-client setting. This contrasts with the results presented by Konečný et al. (2016) for compressed client-to-server updates, where aggressive values of s were admissible. This trend extends to the other compression parameters: server-to-client compression of global models requires much more conservative settings than client-to-server compression of model updates. For example, for CIFAR-10, Konečný et al. (2016) get away with using $s = 0.25$ and $q = 8$ under a random Hadamard transform representation[3]. Meanwhile, in Figure 2.3 we can see that, for the same q and representation, $s = 0.5$ already causes an unacceptable degradation of the accuracy. This is not surprising, since it is expected that the updates' error will cancel out once several of them get aggregated at the server, which is not true for model downloads.

2.3.3 Federated Dropout

We focus on testing how the global model's accuracy deteriorates once we use the strategy proposed in Section 2.2.2. In these experiments, we vary the percentage of neurons (or filters for the case of convolutional layers) that are *kept* on each layer of our models (we call this the federated dropout rate). We always keep the totality of the input and logits layers, and never drop the neuron that can be associated to the bias term.

Figure 2.4 shows how the convergence of our three models behaves under different federated dropout rates. We repeat each experiment 10 times and report the mean among these repetitions. The main takeaway from these experiments is that, for every model, it is possible to find a federated dropout rate less than 1.0 that matches or, in some cases, even improves on the final accuracy of the model.

A federated dropout rate of 0.75 seems to work across the board. This corresponds to dropping 25% of the rows and columns of the weight matrices of fully-connected layers (which

[3]The updates for CIFAR-10 can actually be compressed up to 2 bits.

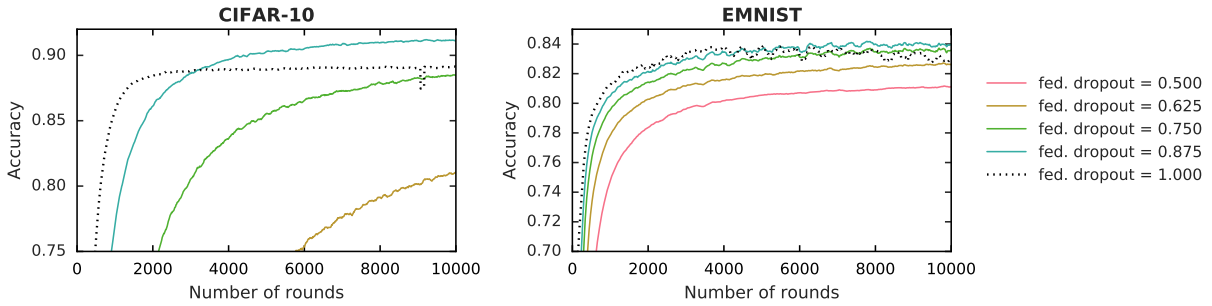

Figure 2.4: Results for *Federated Dropout*, varying the percentage of neurons *kept* in each layer.

translates to a $\sim 43\%$ reduction in size), and to dropping the same percentage of filters of each convolutional layer. Now, because fully connected layers correspond to most of the parameters of the MNIST and EMNIST models, the $\sim 43\%$ reduction will apply to them both in terms of the amount of data that has to be communicated and of the number of FLOPS required for local training. Meanwhile, because our CIFAR model is fully convolutional, gains will be of 25%.

As a final comment, we note that more aggressive federated dropout rates tend to slow down the convergence rate of the model, even if they sometimes result in a higher accuracy.

2.3.4 Reducing the overall communication cost

Our final set of experiments shows how our models behave once we combine our two strategies, lossy compression and *Federated Dropout*, with existing client-to-server compression schemes (Konečný et al., 2016), in order to explore how the different components of this end-to-end, communication efficient framework interact. To do this, we evaluate how our models behave under 3 different compression schemes (aggressive, moderate and conservative) and 4 different federated dropout rates (0.5, 0.625, 0.75 and 0.875). The values for these schemes and rates were picked based on the observed behavior during the previous experiments, being somewhat more conservative as we are now combining different sources of noise. Table 2.3.4 describes the settings for each scheme.

Figure 2.5 shows how our CIFAR-10 and EMNIST models behave under each of the previously mentioned conditions. We repeat each experiment 5 times and report the mean among these repetitions. For all three models, a federated dropout rate of 0.75 resulted in models with no accuracy degradation under all compression schemes except for the most aggressive. For MNIST and EMNIST, this translates into server-to-client communication savings of $14\times$, client-to-server savings of $28\times$ and a reduction of $1.7\times$ in local computation, all without degrading the accuracy of the final global model (and sometimes even improving it). For CIFAR-10, we provide server-to-client communication savings of $10\times$, client-to-server savings of $21\times$ and local computation savings of $1.3\times$.

Based on these results, we also hypothesize that a federated dropout rate of 0.75 combined with a moderate or conservative compression scheme will be a good starting point when setting these parameters in practice.

Scheme	Client-to-Server			Server-to-Client		
	transf.	s	q	transf.	s	q
Aggressive	Kashin's	0.4	2	Kashin's	1.0	3
Moderate	Kashin's	0.5	4	Kashin's	1.0	5
Conservative	Kashin's	1.0	8	Kashin's	1.0	8

Table 2.2: Settings for each of our proposed compression schemes.

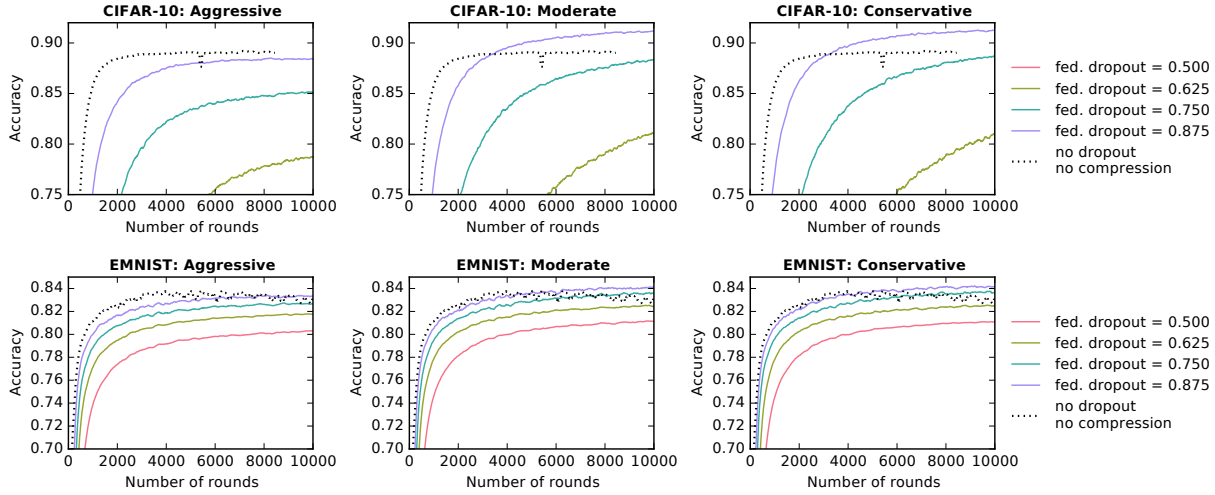

Figure 2.5: Effect of using both compression and *Federated Dropout* on CIFAR-10 and EMNIST.

2.4 Conclusions, Impact and Open Questions

The ecosystem we target is marked by heterogeneous edge networks that can potentially be orders of magnitude slower than the ones in datacenters. At the same time, collaborative learning can be quite demanding in terms of bandwidth, particularly when used to train deep models. We are thus at risk of either restricting the type of models we are able to train using techniques such as FL, or of excluding large groups of users from the collaboration. Both issues are problematic, but because access to high-end networks also appears to be correlated to sensitive factors such as income and age (Anzilotti, 2016; Pew Research Center, 2018), the latter may have implications related to fairness, making it particularly sensitive as we continue the adoption of collaborative systems.

Our work reduces the communication overheads in FL by (1) using lossy compression techniques on the server-to-client exchanges and by (2) using *Federated Dropout*, a technique that only communicates subsets of the global model to each client. Immediate future work can study the effectiveness of other compression techniques in conjunction with *Federated Dropout*, such as natural compression (Horvóth et al., 2022) and sparsification via thresholding (M Abdelmoniem et al., 2021). Future work can also study the effect of adaptively using these strategies (i.e. using more aggressive compression and federated dropout rates for some users) to prevent

unfairly biased models.

Finally, we note that the success of *Federated Dropout* suggests a new avenue of research in collaborative learning in which smaller, perhaps personalized, sub-models are eventually aggregated into a more complex model. Subsequent works that follow this avenue include:

- Horvath et al. (2021) structure the order in which activations are dropped from the layers, ensuring some weights reliably learn across clients. They leverage the systems heterogeneity of the participating devices, dropping a different percentage of activations depending on devices' capacity.
- Guliani et al. (2022) study this avenue in the context of speech recognition models, achieving a size reduction between $6-22\%$ with performance improvements of $34-3\%$. They determine the percentage of neurons to drop in each layer depending on the importance of that layer for learning.
- Alam et al. (2022) use a rolling window to determine which sub-model will be sent in each round. They show their technique works under different amounts of heterogeneity and when clients have diverse systems requirements.

These works suggest that structuring the sub-models results in a better accuracy-communication trade-off over randomly generated sub-models. Moreover, we see that this structure has been explored in ways that depend on the systems requirements, model architecture, and round of communication. Still, an open question remains: whether we can find an optimal structure for the sub-models in a data-dependent manner. Answering this question could help understand the different components of the model, and the different contributions of each collaborator, taking us closer to transparent and modular collaborative systems.

Part II

Explanations

Deriving true clinical utility from models trained on multiple hospitals' data is a key challenge in the adoption of collaborative models in clinical settings. When utility is equated to predictive power, population heterogeneity between centers becomes a key bottleneck in training performant models. Nevertheless, there are other aspects to clinical utility that have frequently been overlooked in this context. Among them, we argue for the importance of understanding how a collaboration may be affecting the quality of a center's predictions. In Chapter 3, we take a step towards deriving this utility by introducing FedeRated CLassifier Selection (FRCLS[4]): an algorithm that reuses classifiers trained in outside institutions and identifies regions of the feature space where collaborators outperform the local center. Most importantly, FRCLS provides interpretable rules to describe these regions of beneficial expertise. Meanwhile, in Chapter 4, we derive further utility from FRCLS by using it to enable the transfer of best practices across healthcare institutions, identifying and supporting opportunities for knowledge transfer.

[4] pronounced as in freckles.

Chapter 3

Understanding Clinical Collaborations Through Federated Classifier Selection

When training Machine Learning (ML) models for healthcare applications, previous studies have shown the advantages of using multiple centers' data. This strategy augments the sample size available for data-intensive models, increases the availability of rare and new events, and potentially improves model performance (Lee et al., 2012; Wiens et al., 2014; Sheller et al., 2018; Curth et al., 2019; Li et al., 2019b). This collaborative approach, however, faces several obstacles that can prevent it from delivering true clinical utility:

- **Limits on data sharing**: Hospitals are responsible for protecting their patients' confidentiality. As such, legal constraints, organizational policies, and ethical barriers often impede centers from sharing patient-level data (Van Panhuis et al., 2014).
- **Heterogeneity**: Data in different centers is inherently heterogeneous. They collect data in different ways, have different laboratory procedures, and have varying care styles and organizational cultures. Of most interest to us, different hospitals serve different populations. As a consequence of this natural population diversity, data from different centers is not identically distributed. However, sharing intelligence across centers, in spite of these misalignments, is a worthwhile effort that can bring improvements in quality of care and reductions in its cost (Lee et al., 2012; Curth et al., 2019). Naive collaborative models will fail to recognize and leverage this variation.
- **Overemphasis on Predictive Power**: ML engineering has traditionally focused on optimizing predictive power, measured through some metric such as empirical accuracy. Nonetheless, clinical utility encompasses other aspects, e.g., eliciting trust in the model through explanations of its predictions or inner mechanisms. In the context of clinical collaborative strategies, we are interested in explaining how the collaboration itself is affecting a center's predictions, e.g., whether a decision is being made based on knowledge from an external center. Concrete rationale of this type can incentivize further cooperation, identify local bottlenecks and inform local resource allocation, or even help identify external best practices.

In this chapter, we introduce `FRCLS`, a classification algorithm that tackles all three obstacles outlined above. To overcome the data sharing obstacle, we adopt the strategy of traditional FL

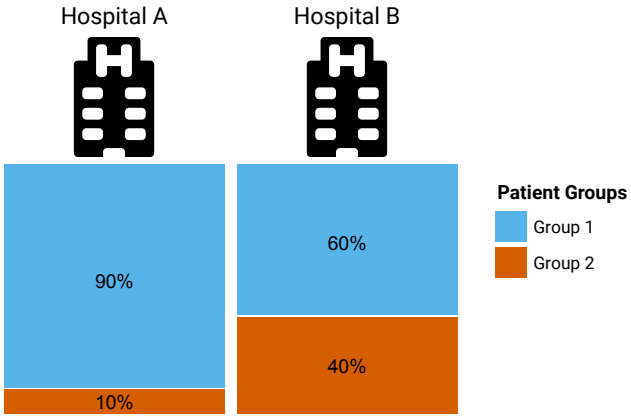

Figure 3.1: Illustration of inter-center population heterogeneity. We hypothesize that a model trained on the data from Hospital B will be a relative expert on patients from Group 2, while a model trained on Hospital A may underperform for that group. It will thus be beneficial to use model B on Group 2 in Hospital A.

techniques: training distributed models through the exchange of their associated parameters instead of the data they are being trained on (Kairouz et al., 2021; Li et al., 2020a; Rieke et al., 2020). To tackle inter-center heterogeneity, FRCLS is designed to adapt its models to the data distribution of a particular hospital, instead of training a single shared model. Finally, FRCLS's focus goes beyond predictive performance, being able to identify groups of patients for which the collaboration is particularly useful.

FRCLS is driven by the intuition that inter-center population heterogeneity makes each hospital an expert on different patient subpopulations. It follows that each center could be an expert in a different region of the feature space, as we illustrate in Figure 3.1. FRCLS leverages this diversity of competence among classifiers and dynamically picks the model that is best for each incoming instance. Other works on FL for healthcare that recognize the challenge of heterogeneity of data address it through techniques such as domain adaptation (Curth et al., 2019; Andreux et al., 2020a) and clustering (Huang et al., 2019). However, they all suffer from the third obstacle above: clinicians cannot judge the utility of the collaboration itself beyond the predictive performance of the resulting models.

We address this last obstacle by explicitly recognizing when FRCLS is leveraging knowledge from an external center. To do this, FRCLS produces rules that clearly delineate the regions of the feature space where external centers are more competent than the local center, providing an interpretable rationale for decisions made by local stakeholders. By optimizing for both accuracy and interpretability simultaneously, and by targeting a deeper understanding of the collaborative predictions, we hope to optimize for our actual goal: clinical utility.

3.1 Related work

We review relevant work in three key directions: (i) Dataset shifts in healthcare, which is a crucial motivation for our work, (ii) FL, to situate our work in the broader context of this field, and (iii) Dynamic Classification, the underlying technique behind FRCLS.

Dataset Shifts in Healthcare. At a high level, dataset shift refers to a scenario in which a ML model is tested on data drawn from a different distribution than the one it was trained on (Subbaswamy et al., 2021). Usual performance guarantees don't apply to this setting, as they refer to how the model will perform on data drawn from the training distribution. Dataset shift may cause the deterioration of clinical prediction models when they are validated on an external distribution and when a collaboratively trained model is applied to individual centers (Lee et al., 2012; Zech et al., 2018; AlBadawy et al., 2018; McKinney et al., 2020). Different works have tried to prevent this degradation by using techniques such as domain adaptation (Curth et al., 2019) and transfer learning (McKinney et al., 2020; Mustafa et al., 2021). FRCLS exploits the model heterogeneity that such a shift causes through its dynamic classifier approach.

On the clinical side, recent efforts to offset dataset shifts include developing and implementing common data models across collaborating institutions. Nevertheless, these are only partially successful, as other considerations, such as local workflows and treatment protocols, are also contributing factors (Matheny et al., 2019).

Federated Learning. Traditional FL algorithms coordinate the training of ML models through a central server, which iteratively broadcasts the current model to participating collaborators and aggregates the updates it receives in return (Li et al., 2020a; Kairouz et al., 2021). Different algorithms vary in terms of how they perform the local updates and the central aggregation (McMahan et al., 2017; Li et al., 2020b; Karimireddy et al., 2020). Previous work has shown the feasibility of using federated techniques of this kind on healthcare problems (Sheller et al., 2018; Li et al., 2019b; Andreux et al., 2020b,a; Caldas et al., 2020), while others have focused particularly on FL's fairness challenges (Li et al., 2020c; Mohri et al., 2019). FRCLS's focus, however, is on the interpretability of its predictions. Its algorithmic approach is also different in that it performs one single exchange between all collaborators instead of several exchanges with a central server.

Dynamic Classification. Our method is related to the Dynamic Classifier Selection (DCS) and the Regression-based Informative Projection Recovery (RIPR) frameworks (Cruz et al., 2018; Fiterau & Dubrawski, 2012), both of which make use of a heterogeneous pool of classifiers and serve a different model for each test instance. Both frameworks select the classifier to be served by estimating the competence of each candidate model on the region of the feature space where the new instance resides. DCS methods estimate the models' performance on the instance's k-nearest neighbors, while RIPR derives a local entropy measure. FRCLS takes a similar approach to the one used by DCS methods, but provides strategies to explaining the selection of the served classifier.

3.2 Federated Classifier Selection

At a high level, FRCLS proceeds in three stages: the local training of classifiers, the exchange of fully trained classifiers between centers, and the dynamic selection of classifiers in each center. We first give a high-level overview of each stage before further detailing the third one, which is the focus of our contributions.

1. **Training of local classifiers**: Each clinical center can independently choose its own type of model, hyperparameter tuning strategy, etc.
2. **Exchange of classifiers**: In this stage, clinical centers exchange both their fully trained classifiers and the local imputation/standardization parameters used during training. We assume an honest-but-curious threat model and thus consider it safe to share this information among clinical centers. A central authority may also anonymize the exact source of the external classifiers. In the end, each hospital is left with a local classifier c_L and a pool of candidate external classifiers $C = \{c_1, \ldots, c_M\}$.
3. **Dynamic selection of candidate classifiers**: This stage takes place at each center independently. The ultimate goal is, for each new incoming instance, to select the most competent classifier among all candidates. We explain the details on how FRCLS does this in Section 3.2.1.

3.2.1 Dynamic selection of candidate classifiers

We are given a local classifier c_L and a set of M external classifiers $C = \{c_m\}_{m=1}^{M}$. Our objective is to determine, for each new instance x, whether to use c_L or one of the elements of C. To make this decision, we set out to quantify the utility of the external classifiers in C relative to c_L. Define

$$L_c(x,k) = \frac{1}{k} \sum_{j \in nn(x,k)} \ell(c(x_j), y_j),$$
$$\rho_m(x,k) = \frac{L_{c_L}(x,k)}{L_{c_m}(x,k)},$$

where $nn(x,k)$ returns indices of the k-nearest neighbors of x, ℓ is the cross-entropy loss, and $c(x)$ is the score that classifier c assigns to x.

Notice that $L_c(x,k)$ estimates classifier c's competence on a given point x by averaging c's loss on the point's k-nearest *known* neighbors. Because it looks at the classifier's loss, L_c is inversely related to c's competence. Meanwhile, $\rho_m(x,k)$ takes the ratio of L_c for the local c_L and an external c_m. Because of L_c's inverse relation to competence, a higher value of ρ_m translates into a higher competence for the external classifier c_m.[1]

We now construct the greedy external classifier c_E which solves

$$c_E(x) = \arg\max_{c_m \in C} c_m(x)$$

[1]For computational stability, the quantity we use in our experiments is $\rho'_m(x,k) = \log \frac{L_{c_L}(x,k)+\epsilon}{L_{c_m}(x,k)+\epsilon}$ for some small ϵ.

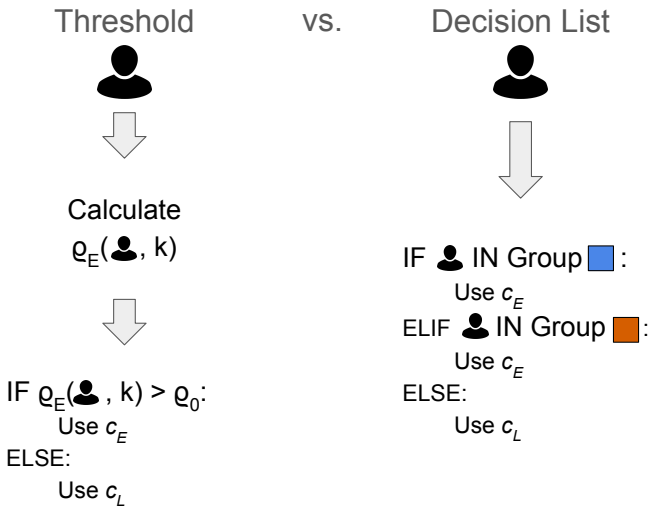

Figure 3.2: Illustration of FRCLS's strategies to select between the local and the external classifiers.

for each new instance x. The final step is to pick between c_L and c_E. A naive strategy would choose c_E whenever $\rho_E(x, k) > 1$. However, this strategy won't be necessarily optimal, as ρ_E quantifies relative competence in terms of loss, which is just a proxy for actual clinical utility. Instead, we propose two data-driven strategies that we illustrate in Figure 3.2.

Competence threshold

Our first strategy finds a threshold ρ_0 such that c_E will be used whenever $\rho_E(x, k) > \rho_0$. We optimize this threshold by minimizing the p-value of a statistical test whose null hypothesis states that c_L's utility is greater than c_E's, where we measure utility in terms of a classifier's correct predictions.

More precisely, we define

$$F(\rho_0) = |\{x_i : \rho_E(x_i, k) > \rho_0, c_L^*(x_i) \neq c_E^*(x_i), c_E^*(x_i) \neq y_i\}|,$$
$$S(\rho_0) = |\{x_i : \rho_E(x_i, k) > \rho_0, c_L^*(x_i) \neq c_E^*(x_i).c_E^*(x_i) = y_i\}|$$

where $c^*(x)$ is the label that classifier c assigns to x. Notice that S is the number of instances where using c_E actually changes the prediction made by c_L, and the new prediction is correct. Meanwhile F is the number of instances that changed predictions to an incorrect one.

Having these quantities, we perform a one-tailed binomial test to check for the statistical significance of $\frac{S(\rho_0)}{S(\rho_0)+F(\rho_0)} < 0.5$. In our experiments, we use a simple grid search strategy to look for the threshold ρ_0 that lets us reject this null hypothesis with the most confidence.

Decision lists

Our second strategy uses c_E if the instance satisfies a set of interpretable rules, and otherwise defaults to c_L. To build these rules, FRCLS uses a rule learning algorithm to create a decision

list that maximizes a lower bound on the mean of ρ_E. Then, we iterate over the list and choose the rule that minimizes the p-value of our binomial test when applied to the instances selected by all rules so far on the list. This will be the last rule that prescribes the use of c_E. Our approach is agnostic to the implementations of the rule learning algorithm. The one we use in our experiments is the one proposed by Moore & Schneider (2002).

3.2.2 Limitations

After we find the threshold or rule that maximizes c_E's utility relative to c_L's, the optimal p-value of our binomial test may still be higher than a satisfying confidence level. In this scenario, both of FRCLS strategies would default to the local model for all instances and there would be no gains in predictive power due to the collaboration. However, the knowledge that the external models do not outperform the local one for any patient subpopulation is still a valuable insight into the limited utility of the collaboration. Additionally, our k-nearest neighbors estimator for L_c is known to suffer from the curse of dimensionality. We also need to tune an extra hyperparameter for it: the number of neighbors or k. We propose a heuristic to perform this tuning in Appendix B.1. We expect other methods, such as cross -validation, to also work.

3.3 Results: Early prediction of sepsis

In this section, we first present the details of our experimental setup before presenting and discussing the results of the local classifiers and FRCLS's competence threshold and decision list strategies.

3.3.1 Experimental setup

We demonstrate our method on an early sepsis prediction task. Sepsis is linked with high mortality, morbidity, and cost of care in hospitalized patients. To mitigate this burden, early identification of risk for sepsis and timely treatment are recommended (Angus et al., 2001). It follows that systems for early and accurate identification of sepsis are of crucial interest for the community (Nemati et al., 2018; Reyna et al., 2019).

Data source. We use the data shared by Reyna et al. (2019) as part of the 2019 PhysioNet/-Computing in Cardiology Challenge. The released data corresponds to ICUs in two geographically distinct hospital systems with different Electronic Medical Record systems. In the rest of the document, we refer to these as hospital system A and hospital system B, matching the nomenclature used by Reyna et al. (2019). The public data accounts for $40,336$ patients and over 1.5 million instances.

ML task. Our ML task is to predict sepsis 6 hours before its onset time, according to the definition used by Reyna et al. (2019). Due to the nature of this task, the label distribution is skewed: only 1.80% of the given labels correspond to the positive class. To facilitate the predictive task and to focus our study on its federated aspects, we randomly undersample the negative labels in order to match the number of negative and positive labels. We are left with

55.8 thousand instances from 20,779 patients. Table B.1 shows the number of instances per hospital system.

Feature choices. We use the features provided by the 2019 PhysioNet Challenge. These consist of a mixture of hourly vital signs, laboratory values, and patient descriptors. Table B.2 and Table B.3 describe the numerical and categorical features provided, respectively. We also use the standard deviations of the mean arterial pressure and the respiration rate (Nemati et al., 2018).

Data splits. We split each hospital system's data into three disjoint sets. First, a training set used for training and tuning the local classifier. Second, a validation set used to either find the optimal ρ_0 or to train FRCLS's decision list. This set is used to measure L_c for all candidate classifiers, i.e., the operator $nn(x, k)$ is restricted to returning instances from this set. Third, we have a test set for estimating FRCLS's performance. We perform the split in a $40/30/30$ fashion.

Predictive models. Our local classifiers are logistic regression models with ridge penalty. We tune the regularization hyperparameter using 5-way cross-validation to optimize for loss. We locally impute missing values, using the mean for numerical features and the mode for categorical ones. Finally, we locally standardize numerical features.

These models are not state-of-the-art for a sepsis prediction task, but they are sufficient for evaluating our method, as our purpose is to selectively use c_E to successfully change local predictions and not to achieve the best possible accuracy on the task.

Evaluation metrics. We are interested in comparing the utility of classifiers c_E and c_L on the instances where FRCLS decides to use c_E over c_L. We measure this relative utility as we did in Section 3.2: by looking at the instances (x, y) for whom $c_L^*(x) \neq c_E^*(x)$. We refer to these predictions as *flipped*, and we consider them as *successful flips* if $c_E^*(x) = y$. If the number of successful flips represents more than 50% of the total number of flips, then we consider the method successful.

Our metric requires crisp predictions from the classifiers. As such, we focus on two constraints that optimize for complementary objectives and allow us to obtain crisp classifiers: holding the true positive rate at 90% (@90% TPR), and holding the false positive rate at 10% (@10% FPR). Given our sepsis prediction task, a center would likely prefer guaranteeing a high recall (@90% TPR), but we show both constraints for the sake of completeness.

Finally, this metric only measures c_E's performance on those instances in which c_E's predictions differ from c_L's. To quantify c_E's performance both when it agrees and disagrees with c_L, we also measure the accuracy of the crisp classifiers on all instances where FRCLS uses c_E over c_L.

3.3.2 Results of local classifiers

We show the performance of our sepsis prediction models trained independently in Hospitals Systems A and B in Figure 3.3. We plot the Receiver Operating Characteristic (ROC) curve for both models when evaluated in the test data of each hospital system. In both cases, c_L either outperforms or matches the performance of c_E, a consistent behaviour throughout the curve. Judging by these results, both hospital systems may have deemed c_E's utility as limited. However, in Section 3.3.3 and Section 3.3.4, we'll show that, by comparing the models'

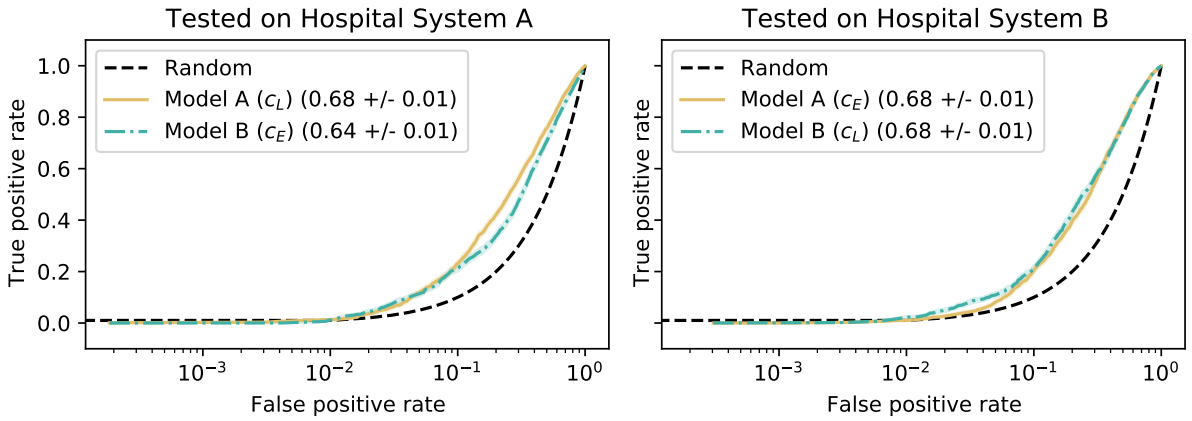

Figure 3.3: AUC ROC for the classifiers when tested on each hospital system. Confidence bands reflect Wilson scores. We plot the false positive rate in logarithmic scale for better visibility of models' performance at clinically relevant low error settings.

local behaviour, these hospital systems do yield utility from their external models c_E, doing so selectively on a subset of their instances.

3.3.3 Results of competence threshold strategy

Table 3.1 presents the results for our competence threshold strategy. For three out of the four scenarios we consider, we obtained an optimal p-value lower than 0.05 in both our validation and test sets, a confidence level we consider satisfying for our experiments. These results speak to FRCLS's generalization ability. We note that, in the one scenario in which FRCLS did not generalize (Hospital System A @90% TPR), the validation set p-value was several orders of magnitude higher than for the other scenarios.

Given our data splits, our test sets consist of over 9.5 and 7.3 thousand instances for hospital systems A and B, respectively. With this strategy, FRCLS ends up using c_E on $15 - 35\%$ of the data. Out of this data, $5 - 18\%$ correspond to successful flips. This translates into the hospital systems reaping real utility out of c_E in most cases, as evidenced in the p-values of our binomial test. Finally, in all cases in which we observe utility in terms of successful flips, we also observe utility in terms of an increase in accuracy.

3.3.4 Results of decision list strategy

To generate our decision lists, we use an implementation of the computationally efficient algorithm proposed by Moore & Schneider (2002). We limit each rule to have a maximum of two features in order to make their interpretation easy, and restrict the minimum support of each rule to at least 1.5% of the validation sample size in each hospital system, to safeguard against overfitting. These hyperparameter settings work well in the presented examples, but they may need to be optimized for other applications.

Hospital System	Val p-value	p-value	Instances handled by c_E	Successful Flips (% of flips)	Local Accuracy	External Accuracy
A (@90% TPR)	1.69e−3	**9.99e−1**	1058	26 (32.91%)	63.71%	61.15%
A (@10% FPR)	3.21e−42	1.52e−9	1525	280 (64.22%)	53.64%	61.77%
B (@90% TPR)	7.26e−31	9.04e−11	1243	195 (68.90%)	49.64%	58.25%
B (@10% FPR)	1.51e−7	1.90e−2	2548	128 (57.14%)	57.50%	58.75%

Table 3.1: Results for our competence threshold strategy for one run. All results are measured on the test set unless specified. In bold, p-values higher than 0.05.

Hospital System	Val p-value	p-value	Instances handled by c_E	Successful Flips (% of flips)	Local Accuracy	External Accuracy
A (@90% TPR)	**2.52e−1**	-	0	-	-	-
A (@10% FPR)	1.12e−6	3.07e−5	1925	299 (58.97%)	57.92%	62.65%
B (@90% TPR)	1.35e−3	1.25e−8	700	134 (70.16%)	51.43%	62.43%
B (@10% FPR)	**8.04e−2**	-	0	-	-	-

Table 3.2: Results for our decision list strategy. Notice that, when *Val p-value* is greater than 0.05 (bolded), no instances are handled by c_E.

Just as in the previous section, we show our results in Table 3.2. Meanwhile, in Figure 3.4, we illustrate the decision lists learned for each of the hospital systems. This time, we encounter two situations in which the optimal p-value in the validation set is lower than 0.05. In these cases, FRCLS defaults to c_L. In the other two situations, Hospital System A @10% FPR and Hospital System B @90% TPR, FRCLS generalizes well, using c_E on $10 - 20\%$ of the test instances, out of which $15 - 19\%$ are successful flips. Compared to our competence threshold strategy, we observe a degradation in our p-values, which are now tens of orders of magnitude greater. This translates into lower ratios of successful flips. This is an expected trade-off, as we can now easily interpret FRCLS's decisions. Conversely, with respect to the previous strategy, the difference in p-values between the validation and test sets has decreased by several orders of magnitude. This is also expected, as the use of a decision list with short rules prevents overfitting.

Finally, we look at the relation between strategies in terms of the instances FRCLS selects to use c_E. We argue that our competence threshold strategy selects two types of instances: some that can be explained with the type of rules we desire, with at most two attributes, and some that are not. Our decision list strategy picks out the former group. In Table 3.3, we show that these easy-to-explain instances correspond to $20 - 40\%$ of the instances selected through competence thresholding.

Using FRCLS's rules

We turn our attention into demonstrating how clinical centers can use the rules learned by FRCLS's decision learning strategy, and into showing possible ways to derive strategic utility from them.

Hospital System A

```
IF PTT > 84.55 AND Phosphate <= 8.42:
    Use c_E
ELIF BUN > 83.56 AND Calcium <= 9.66:
    Use c_E
ELIF Hct > 40.31:
    Use c_E
ELIF Calcium > 9.66:
    Use c_E
ELIF Hgb > 12.14:
    Use c_E
ELSE:
    Use c_L
```

Hospital System B

```
IF BaseExcess > -0.92:
    Use c_E
ELIF FiO2 > 0.65:
    Use c_E
ELSE:
    Use c_L
```

Figure 3.4: Rules learned by FRCLS's decision list strategy for our early sepsis prediction task. Instances satisfying these rules will use c_E instead of c_L.

Hospital System	Set	% Explained by Rules
A (@10% FPR)	Val	40.54%
	Test	38.75%
B (@90% TPR)	Val	21.23%
	Test	20.68%

Table 3.3: Percentage of instances selected by our competence threshold strategy that are successfully explained by our decision list strategy.

In Table 3.4 we show two instances corresponding to two different patients, one from each hospital system. We refer to them as Patient A and Patient B. These are instances for whom the rules of their respective hospital system apply and for whom the use of c_E proves beneficial when holding the appropriate constraints, i.e., $@10\%$ FPR for hospital system A and $@90\%$ TPR for hospital system B, as shown in Table 3.2. In the case of Patient A, FRCLS uses c_E because his hemoglobin is greater than the found threshold. For patient B, it's because his fraction of inspired oxygen is higher than 0.65.

We propose a simple argument to explain why c_E does better than c_L for these two cases: the external hospital center sees relatively more instances that satisfy the relevant rules, i.e., Hgb > 12.14 for hospital system A, and FiO2 > 0.65 for hospital system B. To check for this, we construct a one-tailed z-test to compare the proportion of instances that satisfy the rule in the local and external hospital systems. We call these p_L and p_E, respectively. Our null hypothesis states that $p_L - p_E > 0$. For the two rules in question we obtain p-values of $5.60\text{e}{-4}$ and $4.38\text{e}{-10}$, meaning we have enough evidence to conclude that $p_E > p_L$ in these cases[2].

We recognize that the utility of c_E for a group of patients may be due to different factors,

[2]We note that this z-test shares more information across hospital systems than mere model parameters. However, no raw data is being shared.

Variable	Patient A
Age	69
Gender	Male
BUN	13
Calcium	8.6
Hct	38.5
Hgb	13.1
PTT	28.2
Phosphate	2.5
True Label	1

Variable	Patient B
Age	54
Gender	Male
Base Excess	-3.2
FiO2	1
True Label	0

Table 3.4: Instances whose predictions get flipped with the use of c_E. We present one instance per hospital system. The complete list of features for both instances is shown in Table B.4.

with sample size being just one of many. However, to derive strategic utility, clinical centers do have to explore why others are doing better. If it is just an issue of differing subpopulation sizes, then strengthening institutional cooperation would make obvious sense. However, it could also be due to issues in data capturing, or even differences in the consistency and quality of the clinical practices themselves. Different conclusions could lead to different decisions by stakeholders. We further investigate this question in Chapter 4.

3.4 Extensions

In this section, we briefly describe the work by Potosnak (2022) and Potosnak et al. (2021), which extend FRCLS to learn more robust and interpretable rule.

Motivation. As presented in Caldas et al. (2021b), FRCLS generates decision lists that maximize a lower bound on the mean competence of the external classifier, which we estimate through the proxy variable ρ_E. To obtain this lower bound, the algorithm approximates the distribution of ρ_E as normal, an approximation that falls apart for rules with small support. This phenomenon may lead to FRCLS selecting rules that overfit to the training data, with features that don't correlate with the medical output, or too long to be clinically interpretable.

Method. Potosnak (2022) and Potosnak et al. (2021) propose a rule pruning step using permutation testing and a novel k-nearest neighbor approach to probability estimation. Intuitively, the idea is to discard rules that could not be differentiated from those that would be generated from a dataset where the response variable were randomly shuffled.

Data and ML task. The work uses two different datasets, CH ($1,563$ examples) and MIMIC-II ($1,776$ examples). For each dataset, a different ML task was targeted: prediction of hypotension (CH), and early prediction of mortality (MIMIC-II). Finally, each dataset had two silos: medical ICU (MICU) and surgical ICU (SICU).

Results. Tables 3.5 and 3.6 show some of the results in Potosnak (2022). In particular, they obtain consistently shorter decision lists while still improving over purely local models.

Local Silo	Dataset	Number of rules	
		Without pruning	With pruning
SICU	CH	1.8(1.2)	**1.7(0.5)**
	MIMIC-II	7.8(9.2)	**1.8(1.3)**
MICU	CH	6.8(4.3)	**1.7(0.5)**
	MIMIC-II	14.8(7.2)	**4.2(2.8)**

Table 3.5: Decision list lengths presented by Potosnak (2022) for their method. They present the mean (standard deviation) of 5-fold cross validation.

Local Silo	Dataset	ROC-AUC	
		Local	FRCLS with pruning
SICU	CH	0.875(0.021)	**0.879(0.028)**
	MIMIC-II	**0.851(0.034)**	0.830(0.051)
MICU	CH	0.888(0.012)	**0.893(0.002)**
	MIMIC-II	0.708(0.022)	**0.761(0.050)**

Table 3.6: Mean ROC AUC results from Potosnak (2022) for their rule pruning method. They present the mean (standard deviation) of 5-fold cross validation.

3.5 Conclusions

We proposed an approach to derive clinical utility from Federated Learning (FL) systems that goes beyond an increase in predictive power. Compared to previous works in FL for healthcare applications, we argued for a deeper understanding of potential benefits of the clinical collaborations supported by these systems, particularly of when external knowledge was affecting local predictions, as this understanding can lead to strategic decisions by stakeholders.

We used a dynamic classification framework to contextually leverage models trained at different clinical institutions, and produced simple rules to clearly outline regions of the feature space where one model outperformed the others. We tested our proposed approach on a benchmark sepsis prediction task in two hospital systems, showing that it was capable of providing both a boost in predictive power and interpretable insights into the types of patients most benefited by the collaboration. Such insights can be used to motivate follow up investigations into specifics of clinical practice that may lead to such differences in model performance. These investigations could help identify the most effective practices for industry-wide proliferation, as well as create awareness of potential inefficiencies of organizational culture or processes that may be addressable at local institutions.

Additional research can improve our work. First, designing a feedback mechanism between collaborators could result in further gains: specializing external models to local needs. A more immediate next step is to hybridize both of `FRCLS`'s current strategies, using a competence threshold on those instances not picked out by learned rules. Finally, future work can study the interplay between adding privacy to the shared models, and the rules learned by `FRCLS`.

Chapter 4

Using Machine Learning to Support Transfer of Best Practices in Healthcare

When a set of practices an institution has developed over time is known to systematically lead to positive outcomes, it is an enticing target for implementation at other similar organizations (Berta & Baker, 2004; Tsoukas & Vladimirou, 2001). However, endeavors aimed at transferring such best practices can be inefficient and prone to failure, with common reasons being the difficulty to codify tacit knowledge and lack of adequate motivation for the transfer (Berta & Baker, 2004; Guzman et al., 2015; Elwyn et al., 2007). Just as problematic is the inability to even identify a practice gap between organizations, as it prevents the whole transfer process from taking place. In the context of healthcare, transferring best practices can be done to increase the efficiency and effectiveness of health services, and to improve patient outcomes: goals that healthcare organizations are under continuous pressure to pursue (Berta & Baker, 2004; Perleth et al., 2001). Alas, because of the inherent complexity of healthcare facilities, it is also inherently complex to identify and share best practices among them.

Research into the transfer of best practices in healthcare has been overlooked (Guzman et al., 2015), and thus the community has a limited understanding of which mechanisms could work best in practice. Previous work has focused on theoretically understanding the transfer mechanisms, mostly borrowing tools from other disciplines. For example, Guzman et al. (2015) combined tools from organizational learning and knowledge management with a practice-based perspective, laying down a theoretical framework that aligns the complexity of the practice to be transferred with appropriate transfer strategies. Meanwhile, Berta & Baker (2004) described the types of environments and knowledge where a transfer is possible. Finally, Elwyn et al. (2007) outlined different reasons that could make the transfer process fail in a hypothetical clinical setting using Szulanski (2002)'s sticky knowledge framework.

4.1 Motivating example

In this section, we present an example to motivate our methodology. In this example, we work backwards from a scenario in which FRCLS proves useful, and arrive at the existence of a prac-

Organization	Test instances	Instances handled by c_E	Successful flips (% of flips)	Test p-value
A	1118	111	178 (41.59%)	0.99
B	1883	860	102 (68.00%)	3.94e−7

Table 4.1: FRCLS's results for DRG 291.

tice gap that can explain the apparent utility of the algorithm. We use this to propose a methodology to identify this gap before applying FRCLS.

We consider the task of detecting long lengths of hospital stays in medical inpatient claims. Our claims data comes from the year 2016 Patient Discharge Database (PDD) from California's Office of Statewide Health Planning and Development (OSHPD), and we solve our task using regularized logistic regression models. We define a long stay as one greater than the national average length of stay for the DRG associated with the claim. We perform the analysis separately for each DRG and for two distinct hospital groups, treating each DRG as a different practice and each hospital group as a different organization. We refer to these organizations as Organization A and B. We provide more specific information about our data and ML models in Section 4.3.

For this motivating example, we focus on the DRG 291 associated with heart failure and shock with major complications or comorbidity. The results of using FRCLS for this DRG are shown in Table 4.1. These results indicate an asymmetrical relation between Organizations A and B in terms of knowledge sharing: Organization B observes significant gains from using Organization A's model, but the reverse is not true. In light of this asymmetry, it is natural to inquire about possible causes of the apparent discrepancy. We explore three hypotheses that could explain it:

- **Difference in sample sizes**: If Organization A were to have a larger data sample to train its model than Organization B, then we could explore the possibility that the observed asymmetry is due to this difference. However, for DRG 291, Organization B has 20% more of these claims than Organization A, as seen in Table C.1. As a result, we discard this hypothesis.
- **Difference in model performance**: An inherent difference in model performance could explain the asymmetrical relationship observed in Table 4.1. If this hypothesis were to hold, we would expect the model trained in Organization A to outperform the model trained in B when evaluated on B's data, and we should observe similar effects when evaluating these models on A's data. In Figure 4.1, we see however that the Receiver Operating Characteristic (ROC) curve performances of these models are within each other's error bands. For a more detailed analysis, we focus on comparing the true negative rates at a fixed 90% true positive rate and construct 95% confidence intervals for the difference in performance between models. Both intervals, which are plotted later on in Figure 4.4 (left), end up containing 0, further indicating lack of evidence to support the hypothesis that the observed trade-off in utility of these models can be attributed to difference in their predictive power.
- **Difference in consistency and quality of care in patient treatment practice**: Having discarded perhaps the most obvious candidate explanations for the observed discrepancy, we

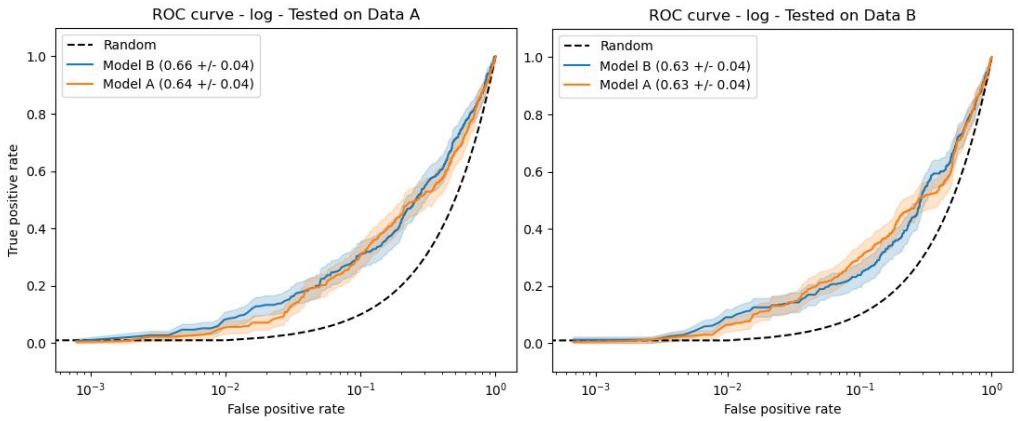

Figure 4.1: ROC curves for our motivating example. We plot the false positive rate in logarithmic scale for visibility.

turn our attention to the actual practice being modeled. With the available data we cannot directly evaluate consistency of clinical decisions, but we can rely on a proxy: the entropy of the distribution of scores produced by each classifier. We consider a higher entropy to be indicative of a more ambiguous practice in an organization - this happens when a model trained on operational data from this organization yields predictions that are less crisp at discriminating, for example, long and short stays in the hospital (to refer back to our working example). Such differences in the apparent crispiness of the decision making process between organizations can exist even if the predictive performances of the models trained to automate such decisions do not differ when measured with common means such as ROC analysis.

We plot the score distributions for both models, evaluated on data from Organization B, in Figure 4.2. Here, we see that the entropy resulting from model A applied to data in B is indeed lower than the one resulting from applying model B to its own test data. We check that this difference is statistically significant by constructing 95% confidence intervals for the differences in entropy, which we'll plot later on in Figure 4.4 (right).

Not having insight into actual operations of each organization, we cannot definitely conclude that there indeed exists a practice gap by just using medical claims records to support such judgement. But we propose that the presented result could motivate a thorough investigation and, if warranted, a properly designed intervention in organizations that show an opportunity for improving crispiness of their practice, unless additional evidence could provide simpler explanations for the results in Table 4.1.

In our proposed methodology, we are interested in first identifying the apparent practice gaps. These gaps will motivate organizational audits and adjustments to existing processes to align performance of the target organization up to the apparent levels of their peers demonstrating a more consistent, crispier decision-making practice. Such organizational change may be, however, effortful and time consuming to prepare and implement. As a result, we also propose using `FRCLS` as a stop-gap solution. It will selectively apply the crispier decisions from a better-practice organization when assessing an opportunistically selected subset of all cases for

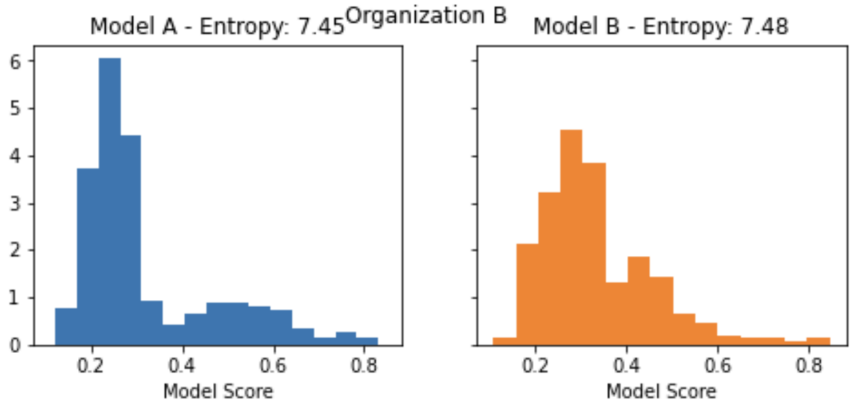

Figure 4.2: Score distributions for our motivating example. Model A is more confident in its predictions than model B when evaluating both models at Organization B. This is reflected in a lower entropy for model A.

which the external model is expected to confidently outperform the model trained on the local data. This strategy aligns with the transfer process steps outlined by Elwyn et al. (2007), and it mitigates or at least postpones the effort of implementing a solution that may not be necessary if such algorithmic work-around is acceptable.

4.2 Identifying practice gaps

Given a machine learning task, we propose that a well-performing and confident model is indicative of a robust and consistent practice. Consequently, if a model trained in an external organization shows a greater performance or confidence than a locally trained model, this suggests that there is a practice gap between the institutions. The models should be evaluated on the same set of data for conclusions to be valid. As such, we could imagine that organizations share their trained classifiers, perhaps through an arbiter, and then test and evaluate external models on their own local data. This siloing is a restriction often imposed in the healthcare industry (Van Panhuis et al., 2014).

We outline two possible scenarios that can occur when comparing two models, one local and one external, on data from one organization of interest:

- The case in which one model dominates in terms of some performance metric (for example, Area Under the ROC Curve (ROC AUC), or recall rate at a fixed low false positive rate) is straightforward: if the local model dominates, no adverse gap exists; if the external model dominates, there is a possible practice gap. The latter case, however, may not be very common in reality due to the usual degradation of clinical models whenever they are used outside of the institutions they were trained in (Siontis et al., 2015).
- The case where the models are similarly well-performing, but show a different degree of confidence in their predictions. In this case, if Organization B produces a more confident model than A, as measured on data from A, then this could be indicative of a more consistent

practice implemented in organization B. This is the case we encounter the most often in our experiments. To tackle these situations, we quantify model confidence through the entropy of the distribution of scores generated by the model. This is the same proxy we used in our motivating example to quantify consistency of practice. Note that alternative proxies of quality of practice can also be used instead without changing the proposed framework.

Regardless of the scenario that led to the identification of the potential practice gap, it is important to verify that this conclusion is not the product of statistical artifacts. In particular, one should be aware of whether the organizations have different amounts of data to work with. More data is usually associated with better-performing models, so it is crucial to control for this effect before making any recommendations that will consume resources and impact organizations in the long-term.

4.3 Results: Detection of overly-long hospital stays

Data description. We use the data from the year 2016 Patient Discharge Database (PDD) from California's Office of Statewide Health Planning and Development (OSHPD). This dataset consists of over 3.8 million inpatient medical claims, with associated information about the diagnoses and procedures reported in each claim. We limit our study to the top 20 diagnosis related groups (DRGs) with the highest volume of claims, which we model separately as different practices that could be improved. Out of these, we exclude DRGs relevant to newborns to further focus our analysis.

Moreover, for clarity of presentation, we focus on data pertaining to only two hospital groups. Nevertheless, our analyses can be directly applied to larger numbers of organizations, and to entities at different levels of organizational granularity, for instance individual hospitals or practices within a hospital system. Throughout the rest of this text, we refer to these selected hospital groups as Organization A and Organization B. We are left with 14 DRGs representing over 240 thousand inpatient claims. Table C.1 shows descriptions of the considered DRGs as well as sample sizes for each DRG in each organization.

Machine learning models. To demonstrate our method, we propose a machine learning task that predicts whether a claim will be associated with an unusually long length of hospital stay. To fulfill this task, we binarize the traditional length of stay prediction task, using the national average length of stay of each DRGas a threshold to determine the labels: a positive label represents a stay longer than the national average while a negative label represents a shorter stay. We consider a scenario in which our defined organizations could not share their data, replicating the realistic setting in which institutions are constrained by patient confidentiality. We end up with one model per DRG, per organization.

We solve the task using regularized logistic regression, but our methodology admits any type of model with an ability to estimate loss of predictions made on individual query data instances. We use 5-way cross-validation to pick the regularization hyperparameter, performing a grid-search over $1,000$ settings. We train our models using 70% of available data, and report our results on the remaining 30% held out for testing. We use the true negative rate at a fixed 90% true positive rate (TNR @ 90% TPR) as our evaluation metric. This performance

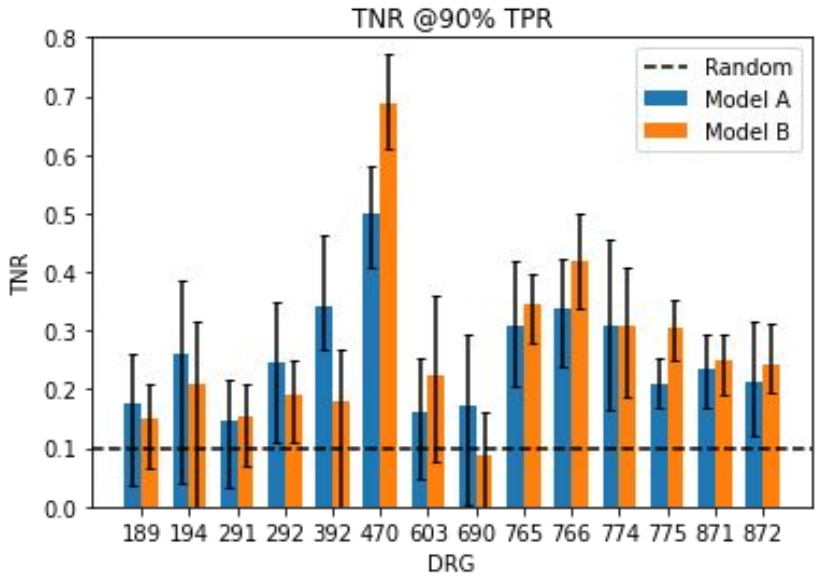

Figure 4.3: TNR @ 90% TPR for the trained models. Each model is tested on data from the organization where it was trained. The error bars reflect 95% bootstrap confidence intervals, for which we performed $1,000$ resamplings. Note that except for a couple of DRGs, results of these models on their own data are not statistically discernible.

indicator quantifies the desired trade-off between predicting lengths of stay in a normal range while maintaining high sensitivity, but alternative performance metrics can surely be used in its stead as well. We present the performance of the resulting models in Figure 4.3. We refer to the model trained at Organization A as model A, and similarly for model B.

Identification of practice gaps. We begin by observing the differences in performance between model A and model B once both models are used in the same organization. We plot these differences in Figure 4.4 (left). We observe that most of our confidence intervals include zero difference, and in these cases we cannot claim the prevalence of one model over another. We see only two intervals that do not contain zero: DRG 392, Organization A and DRG 765, Organization B. Both of these are composed exclusively of positive differences, which indicates that the local model outperforms the external. For these cases, we have enough evidence to discard a possible existence of a practice gap, as it is defined in our approach.

Next, we proceed to identify models that are similarly performant on data from a given organization, but show a difference in entropy of the distribution of their prediction scores. Figure 4.4 (right) shows these results. Here, we are interested in intervals composed exclusively of positive differences, as they indicate that the entropy of the local model is greater than the entropy of the external one, and a sought-after gap may indeed exist. Table 4.2 summarizes the DRGs and organizations where we have identified potential practice gaps.

Inspection of statistical artifacts. As previously mentioned, we need to verify that the potential gaps we identified are not products of statistical artifacts. In particular, we want to make sure that they are not by-products of the differing sample sizes between organizations. To do this, for each DRG, we subsample the training dataset in the organization with the most

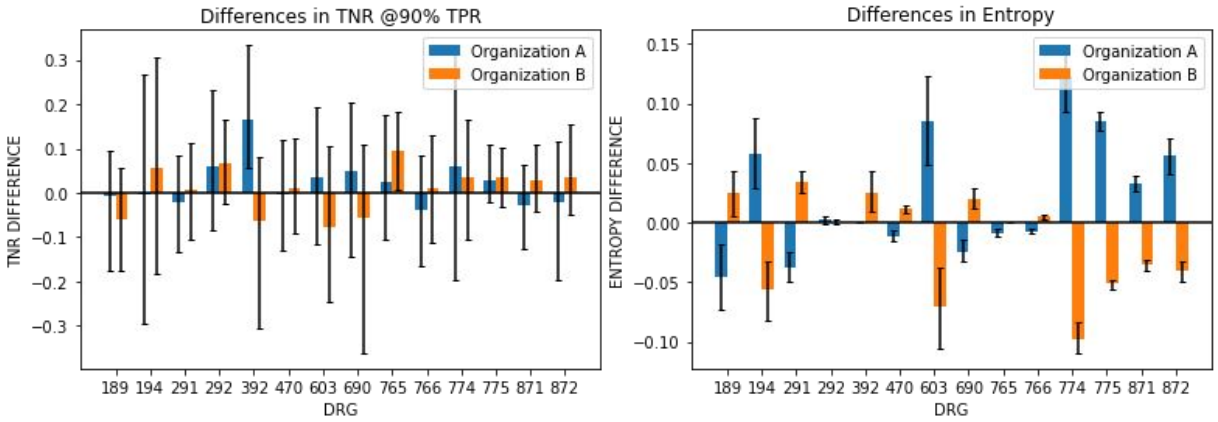

Figure 4.4: Differences in performance (left) and entropy (right) when model A and model B are used in the same organization. The presented differences correspond to the the local model minus the external one. Error bars correspond to bootstrap confidence intervals. We maintain an overall confidence coefficient of 95%, using Bonferroni's method to correct for making multiple comparisons.

DRG	189	194	291	292	392	470	603	690	765	766	774	775	871	872
A	E	⋆	E	E	P	E	⋆	E	E	E	⋆	⋆	⋆	⋆
B	⋆	E	⋆	E	⋆	⋆	E	⋆	P	⋆	E	E	E	E

Table 4.2: Conclusions before inspecting for statistical artifacts. A star ($\star$) indicates a potential practice gap, as defined by our framework. A "P" indicates the evidence to discard the potential gap came from the difference in model performance, while an "E" indicates the evidence came from the difference in score entropies.

data until it has the same volume as the other organization. We repeat this random subsampling process ten times and train a completely new model each time. The results of this exercise are shown in Figure 4.5.

Controlling for size, we see some of our conclusions change: some performance differences we thought were negligible, i.e., the confidence interval contained zero, turn out not to be. This happens for Organization A, DRGs 603, 774 and 775. In these cases, our confidence intervals are composed exclusively of positive differences. We take this as evidence to discard a practice gap.

FRCLS **Evaluation.** Table 4.3 shows our results for the DRGs and organizations where we identified a practice gap. We pay close attention to the columns Train p-value and Test p-value. They both relate to how confident we are that the percentage of correct flips is lower than 50%, but the latter is only available after evaluation. It follows that if Train p-value is too large, we will be discouraged from using this framework at test time, recommending the use of only the local model instead. This happens for three of our identified scenarios: DRG 470, 690 and 766, all for Organization B. As a confidence threshold we used an alpha of 0.05 and corrected for multiple comparisons using Bonferroni's method. For all other scenarios, we observe both a

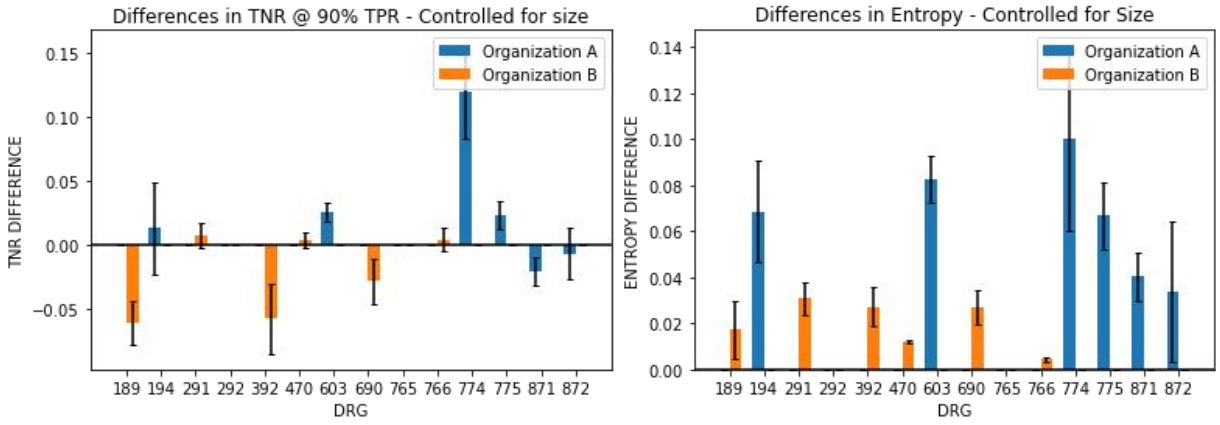

Figure 4.5: Differences in performance (left) and entropy (right) when model A and model B are used in the same organization. The presented differences correspond to the the local model minus the external one. Error bars correspond to bootstrap confidence intervals. We maintain an overall confidence coefficient of 95%, using Bonferroni's method to correct for making multiple comparisons.

statistically significant Train p-value and Test p-value. This means we are both confident about using the method and have enough evidence to claim the method is beneficial on our test data.

4.4 Discussion

We discuss our results on the inpatient claims dataset and our limitations.

ML-aided practice transfer for inpatient claims. Table 4.3 enumerates the nine DRGs and organizations where our methodology identified a potential practice gap. If these organizations were to undertake transfer processes for these practices, FRCLS could support six of these processes, allowing the local organization to reap benefits from the external knowledge, codified through the external model, sooner. In the other three cases (DRG 470, 690 and 766, all for Organization B), our suggested solution recommended using only the local model. All these three cases had a difference in entropy that suggested a more ambiguous local practice, but had some of the lowest absolute differences identified, as observed most clearly in Figure 7. A more robust approach could establish a threshold for the difference in entropy, but this threshold may end up being application dependent.

Knowledge gaps as practice gaps. Our exercise points towards a relationship between the gaps found by FRCLS, which we refer to as "knowledge" gaps, and "practice" gaps between institutions. In rough terms, we showed that the knowledge gap could act as a proxy for an identified practice gap. However, we cannot establish complete equivalence as the scope of this study does not allow us to fully observe the best practice being modeled.

Limitations. There exist even further possible differences between institutions, besides those explored in this chapter, that can explain apparent practice gaps, e.g., differing ML capabilities, in terms of the adoption and incorporation of ML practices. Future research efforts

DRG	Org.	Train p-value	Test instances	Instances handled by c_E	Successful Flips (% of flips)	Test p-value
189	B	$1.03\text{e}{-8}$	1,561	791	84 (78.50%)	$1.20\text{e}{-9}$
194	A	$1.30\text{e}{-45}$	595	594	329 (81.43%)	$2.47\text{e}{-39}$
291	B	$3.94\text{e}{-7}$	1,883	860	102 (68.00%)	$6.15\text{e}{-6}$
392	B	$5.61\text{e}{-5}$	1,353	501	54 (65.06%)	$4.02\text{e}{-3}$
470	B	0.86	6,305	-	-	-
690	B	0.87	527	-	-	-
766	B	0.30	3,013	-	-	-
871	A	$2.42\text{e}{-67}$	4,116	3,869	886 (73.59%)	$1.11\text{e}{-62}$
872	A	$7.28\text{e}{-94}$	1,570	962	402 (78.51%)	$2.17\text{e}{-40}$

Table 4.3: Results of FRCLS as a stop-gap solution.

should be made to develop and test methods that differentiate between differences in these axes and practice gaps. Future work can also validate the generalizability of the proposed approach by extending it to new multi-source datasets. A more ambitious next step would, ideally, observe the identified practices and validate both the existence of a gap and the utility of the stop-gap solution. Such a research study could be impractical, requiring coordination from multiple institutions and the undertaking of an actual transfer process, which is resource-intensive. Another avenue with a more restricted scope is the identification of more desirable model qualities correlated with best practices attributes that have been identified in knowledge management work, beyond high performance and low entropy.

Part III

Expert Supervision

Learning from on-device data has enabled intelligent mobile applications ranging from smart keyboards to apps that predict abnormal heartbeats. However, due to the sensitive and distributed nature of this data, expert annotations are often unavailable. Consequently, existing federated learning techniques that learn from on-device data mostly rely on unsupervised approaches, and are unable to capture expert knowledge via data annotations. In this chapter, we explore one specific way to codify this expert knowledge: using programmatic weak supervision, a principled framework that leverages *labeling functions* (i.e., heuristic rules) in order to label vast quantities of data, without direct access to the data itself. We introduce Weak Supervision Heuristics for Federated Learning (WSHFL[1]), a method that interactively mines and leverages labeling functions to annotate on-device data in cross-device federated settings. Our experiments across two data modalities demonstrate that WSHFL achieves competitive performance compared to a fully supervised baseline while reducing the need for direct data annotations.

[1] pronounced as in wishful.

Chapter 5

Encoding Expert Knowledge Into On-Device Data Using Weak Supervision

Learning from on-device data has the potential to enable increasingly intelligent mobile applications (McMahan et al., 2017): from smart keyboards that boost usability (Hard et al., 2018) to health apps that improve patient outcomes (Fitzpatrick et al., 2017; Bui & Liu, 2021). Nevertheless, due to its sensitive and distributed nature, on-device data cannot be annotated by external experts (Wang et al., 2021). Thus, previous efforts to train models on this type of data have mostly relied on unsupervised methods (Hard et al., 2018; Lu et al., 2021) or have used user contextual signals as supervision (Yang et al., 2018). However, for some critical applications, these approaches may fall short.

As a motivating example, consider training an arrhythmia detection model using electrocardiogram (ECG) data generated in smart watches. This task requires both respecting the sensitive nature of the data and capturing clinicians' expertise, e.g., via annotations of the ECGs. In this work, we consider federated learning methods to accomplish the former: keeping the data isolated on-device and instead exchanging model parameters (McMahan et al., 2017; Wang et al., 2021). However, the question of how to capture the clinicians' expertise into the federated model is an active area of research (Jeong et al., 2020; Liu et al., 2021; Zhuang et al., 2021; Wu et al., 2021).

In this chapter, we explore a particular strategy for codifying this expert knowledge: using LFs, functions that assign imperfect labels to subsets of the data and that can be used to automatically label training data (Ratner et al., 2017; Rühling Cachay et al., 2021). Encoding supervision through LFs is referred to as Programmatic Weak Supervision (PWS) (Ratner et al., 2016; Zhang et al., 2022), and it has had success in centralized settings (Fries et al., 2019; Dunnmon et al., 2020; Goswami et al., 2021; Dey et al., 2022). To the best of our knowledge, PWS has not been explored in federated scenarios, where the focus for encoding expert supervision has been on semi-supervised and self-supervised approaches (Jeong et al., 2020; Liu et al., 2021; Zhuang et al., 2021; Wu et al., 2021).

We introduce Weak Supervision Heuristics for Federated Learning (WSHFL), a method for mining and leveraging LFs in a cross-device federated setting. WSHFL proceeds in two stages:

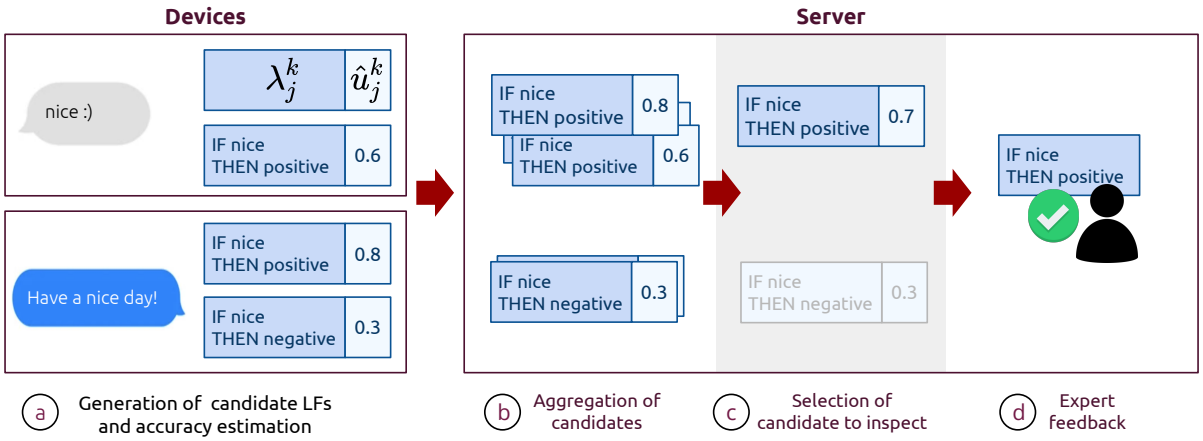

Figure 5.1: Visualization of WSHFL's strategy for generating LFs. Based on on-device data, (a) candidate LFs are generated alongside an estimate $\hat{u}$ of how probable an expert would find them useful. These candidates and estimates are then sent over to the server, where they are (b) aggregated before (c) one candidate is selected to be inspected by an expert. This (d) expert feedback is then used to generate future estimates $\hat{u}$.

1. **Mining of LFs (or heuristics)**: WSHFL automates the crafting of LFs (Varma & Ré, 2018; Boecking et al., 2020), incorporating expert feedback on which ones they consider useful. Only parameterized LFs are exchanged while the data is kept isolated on-device. Figure 5.1 presents a brief overview of this strategy.
2. **Training of the PWS model**: Once we have a set of LFs, we can use them to train a PWS model. WSHFL leverages the architecture proposed by Rühling Cachay et al. (2021), training it in a federated manner.

We argue that the main challenge of adopting PWS into cross-device federated methods is the crafting of LFs. In practice, crafting these functions is a data-dependent process, as experts rely on available data in order to extract dataset-specific heuristics (Varma & Ré, 2018; Boecking et al., 2020; Zhang et al., 2022). In federated learning, however, experts cannot freely explore the on-device data. To tackle this obstacle, we extend the work of Boecking et al. (2020) in automatic generation of LFs to the particulars of cross-device federated learning.

The contributions of this chapter are as follows:

1. We introduce PWS into the federated setting, with the objective of encoding expert knowledge into federated models through their inspection of candidate LFs that are mined from the on-device data. To this end, we propose approaches for two components of a standard PWS workflow in a federated set-up: the generation of candidate LFs, and the training of a model given LFs selected by the expert (Zhang et al., 2022).
2. We conduct experiments on three datasets across two data modalities, demonstrating the feasibility of our approach compared to a fully supervised baseline. We also investigate each of our components, demonstrating their independent utility.

```
def nice_lf(review):
    if "nice" in review:
        return POSITIVE
    else:
        return ABSTAIN
```

Figure 5.2: Example of a labeling function. If the unigram "nice" appears in a review, then it votes for the positive class, otherwise it abstains from voting.

3. Our work is amongst the first to learn classification models from unlabeled distributed time-series data. Previous similar work has assumed access to labels (Zhang et al., 2020; Xu et al., 2021; Choudhury et al., 2019), while we only consider expert supervision over LFs.

5.1 Related Work

Programmatic Weak Supervision. Programmatic weak supervision (PWS) has been proposed as an alternative framework to the expensive and time-consuming process of point-by-point labeling used for supervised machine learning. PWS leverages multiple sources of noisy supervision, expressed as LFs, to label large quantities of data (Zhang et al., 2022). LFs, such as the one presented in Fig. 5.2, are imperfect and may generate conflicting labels on certain data points. Thus, a *label model* (Ratner et al., 2016; Rühling Cachay et al., 2021) is used to aggregate the noisy votes of labeling functions into training labels. These labels are then used to train an *end model*, which learns to generalize the relationship between features and the learned labels. Recent studies have also explored end-to-end approaches that couple the label and end models, leading to state-of-the-art performance (Rühling Cachay et al., 2021). To the best of our knowledge, the PWS literature has only focused on centralized settings.

Automatic Mining of LFs. Hand-crafting LFs requires expertise and data exploration, which can be resource-intensive (Boecking et al., 2020; Varma & Ré, 2018). To address this, previous methods have aimed to automate the creation of LFs given some extra supervision such as seed LFs (Li et al., 2021), labeled data (Varma & Ré, 2018; Awasthi et al., 2020), class descriptors (Gao et al., 2022), or instance-wise expert feedback (Nashaat et al., 2020). Boecking et al. (2020) introduce Interactive Weak Supervision (IWS), an algorithm that learns useful heuristics from user feedback at the LF level. WSHFL leverages this particular type of expert supervision while tackling challenges inherent to the federated scenario.

Semi-supervised and Self-supervised methods in Federated Learning. We can also codify expert knowledge into federated models by using self-supervised and semi-supervised learning. Recent works that have studied this alternative rely on a centralized dataset available for annotation by the experts, and augment the federated learning procedure with techniques such as consistency regularization (Jeong et al., 2020; Liu et al., 2021) and contrastive learning (Zhuang et al., 2021; Wu et al., 2021). However, these techniques usually rely on the ability to augment their data at scale, and have thus been mostly used on image data (Jeong et al., 2020; Zhuang et al., 2021; Wu et al., 2021; Liu et al., 2021).

Time-series federated learning with expert supervision. Federated learning from time-series data is an active area of research (Ding et al., 2022). Nevertheless, prior work on federated learning with time-series is limited to unsupervised classification such as anomaly detection (Liu et al., 2020; Huong et al., 2021), regression (Brophy et al., 2021) and forecasting (Tonellotto et al., 2021). While some studies have considered supervised classification in a cross-silo setting, they assume access to labels with a primary emphasis on privacy preservation (Zhang et al., 2020; Xu et al., 2021; Choudhury et al., 2019).

5.2 Weak Supervision Heuristics for Federated Learning

5.2.1 Problem Formulation

We aim to train an end model f from unlabeled data distributed across devices or clients, as they are commonly called in federated learning. These clients communicate with a server that has no access to the clients' data and orchestrates training. We assume stateless clients as is the norm in cross-device federated learning.

Inputs: For each client k, let $(x^k, y^k) \sim \mathcal{D}^k, \mathcal{D}^k \sim \mathcal{P}$ be the data generating distribution, where $x_i^k \in \mathcal{X} = \mathbb{R}^d$ and the labels belong to one of C classes: $y^k \in \mathcal{Y} = \{1, \ldots, C\}$. As is common in the federated setting, we assume the data between clients is not identically distributed, but all clients share the same feature and label space, *i.e.* $\forall k,\ x_i^k \in \mathcal{X}, y^k \in \mathcal{Y}$. Each client only observes a sample $X_k = \{x_i^k\}_{i=1}^{n_k}$ of n_k unlabeled data points. We also have access to an expert located at the server who is able to determine the utility of a given LF. In Section 5.2.2, we formalize a notion of utility.

Goals: Our ultimate goal is to collaboratively train an end model $f : \mathcal{X} \to \mathcal{Y}$ using unlabeled data from all clients and expert feedback at the server. To this end, WSHFL first uses the distributed data to generate candidate LFs $\lambda = \lambda(x) \in \{0\} \cup \mathcal{Y}$, where 0 means that the LF abstained from labeling any class. Then, WSHFL identifies a set of useful LFs $\mathcal{L}^*$ based on the expert's feedback (Boecking et al., 2020). Finally, WSHFL uses $\mathcal{L}^*$ to train a PWS model on the clients' data, obtaining the resulting end model f.

5.2.2 Automatic Mining of LFs

In this step, WSHFL sequentially shows candidate LFs to the expert at the server. In each step t, the expert inspects a given candidate λ_t and assign it a label $u_t \in \{0, 1\}$ corresponding to whether they believe its accuracy

$$\alpha_t = P(\lambda_t(x) = y | \lambda_t(x) \neq 0)$$

is better than random, i.e., $\alpha_t > 0.5$. This step finally returns those LFs that the expert believed were accurate: $\mathcal{L}^* = \{\lambda_j \in Q_T : u_j = 1\}$.

Algorithm 1: WSHFL mining of labeling functions

Input: Number of expert queries T, seeds S.

1 $Q_0 \leftarrow S$
2 **for** $t = 1, \ldots, T$ **do**
3 $\quad \lambda_t \leftarrow \texttt{FederatedAcquisition}(Q_{t-1})$
4 $\quad u_t \leftarrow \texttt{ExpertQuery}(\lambda_t)$
5 $\quad Q_t \leftarrow Q_{t-1} \cup (\lambda_t, u_t)$

Output: $\{\lambda_j \in Q_T : u_j = 1\}$

6 **Function** `FederatedAcquisition(`Q`)`
7 $\quad \mathcal{L}_0 \leftarrow \emptyset$
8 $\quad$ **for** $r = 1, \ldots, R$ **do**
9 $\quad\quad$ Select K clients at random.
10 $\quad\quad$ **retrieve from each client**
11 $\quad\quad\quad \mathcal{L}_k \leftarrow \texttt{TrainClient}(Q)$
12 $\quad\quad \mathcal{L}_r = \mathcal{L}_{r-1} \cup \bigcup_{k=1}^{K} \mathcal{L}_k$
13 $\quad \mathcal{L}' \leftarrow \texttt{Aggregate}(\mathcal{L}_R)$.
14 $\quad \lambda \leftarrow \texttt{SelectBest}(\mathcal{L}')$.
15 $\quad$ **Return** λ
16 **Function** `TrainClient(`Q`)`
17 $\quad$ Train neural network $h_k : \tau_k(\lambda) \rightarrow u$ using Q.
18 $\quad$ Generate candidate LFs $\mathcal{L}_k = \{\lambda_j^k\}_{j=1}^{p_k}$.
19 $\quad$ Use h_k to estimate $\hat{u}_j^k$ for $\lambda_j^k \in \mathcal{L}_k$.
20 $\quad$ **Return** $\{(\lambda_j^k, \hat{u}_j^k)\}_{j=1}^{p_k}$

Algorithm 1 describes our general procedure. The main challenges to highlight are (1) the generation of candidates LFs in a federated scenario (lines 18 and 13), and (2) the selection of the LFs we show to the expert (line 14).

Generation of candidate LFs

To generate candidate LFs in a federated setting, WSHFL leverages two domain specific processes:

- A client process that takes the unlabeled data $\{x_i^k\}_{i=1}^{n_k}$ and produces candidate heuristics $\mathcal{L}_k = \{\lambda_j^k\}_{j=1}^{p_k}$ in each individual client. See line 18 in Algorithm 1.
- A server process that aggregates similar candidates proposed across clients into G LFs $\mathcal{L}' = \{\lambda'_j\}_{j=1}^{G}$. See line 13 in Algorithm 1.

Notice that a parameterization of the $\mathcal{L}_k$ generated at the clients is shared with the server. In Section 5.3, we describe this parameterization, as well propose different generation and aggregation methods for the two data modalities we work with.

Selection of next LF to inspect

We cast this task as an *active search* problem (Boecking et al., 2020), where we sequentially inspect data (candidates) in order to discover members of a desired class ($u_j = 1$) (Garnett et al., 2012). Thus, at time step t, we require access to the posterior probability $P(u = 1|\lambda, Q_{t-1})$, where $Q_{t-1} = \{(\lambda_j, u_j)\}_{j=1}^{t-1}$ correspond to the previously inspected candidates and their expert feedback.

To estimate this probability, WSHFL trains a model $h_k : \tau_k(\lambda_j) \rightarrow u_j$ in each client that predicts u_j given the client-specific representation

$$\tau_k(\lambda_j) = (\lambda_j(x_1^k), \ldots, \lambda_j(x_{n_k}^k)),$$

for all elements of Q_{t-1}. This model is then used to obtain estimates of $\hat{u}_j^k = h(\tau_k(\lambda_j^k))$ for the candidates that the client generates, which are shared with the server alongside the proposed candidates. See function `TrainClient` in Algorithm 1.

When WSHFL aggregates similar candidates at the server into $\lambda' \in \mathcal{L}'$, it also aggregates the accuracy estimates $\hat{u}_j$ of the candidates being aggregated, treating them as sample estimates of $P(u' = 1|\lambda', Q_{t-1})$. More concretely, let A be the collection of candidates being aggregated, to estimate our posterior probability, we use a $1 - \delta$ lower confidence bound on the mean

$$P(u' = 1|\lambda', Q_{t-1}) = \frac{1}{|A|} \sum_{j \in A} \hat{u}_j - \sqrt{\frac{\log(\frac{2}{\delta})}{2|A|}}.$$

We use a lower bound to account for the high variance of the simple mean when we are aggregating a low number of candidates. Finally, we use a one-step look-ahead search strategy, picking the aggregate $\lambda' \in \mathcal{L}'$ with the highest $P(u' = 1|\lambda', Q_{t-1})$.

5.2.3 Training of the PWS Model

Once we have $\mathcal{L}^*$, we can use these LFs to train label model g and the resulting end model f on the clients' unlabeled data. In this work, we leverage the Weakly Supervised End-to-end Learner (`WeaSEL`) proposed by Rühling Cachay et al. (2021), a state-of-the-art PWS model. Like most PWS models, `WeaSEL` was proposed for centralized data. However, it's architecture makes it amenable to be learned in a distributed setting.

The key idea of `WeaSEL` is to use a two-player cooperative game between two models with different views of the unobserved label through the lens of the features and the LF votes, minimizing a pair of objectives of the form

$$L_f(\mathcal{D}) = \mathbb{E}_{\mathcal{D}}[L(y_f, \texttt{stop-grad}(y_g))] \quad \text{and} \quad L_g(\mathcal{D}) = \mathbb{E}_{\mathcal{D}}[L(y_g, \texttt{stop-grad}(y_f))]$$

where L is a noise-aware loss (*e.g.*, cross-entropy), $y_f = f(x)$ and $y_g = P(y|\lambda)$ are probabilistic labels generated by the label model g that takes both features x, LF outputs $\lambda(x)$ and class balances $P(y)$ as input. To intuitively understand this game, first assume that the probabilistic labels supplied by g are accurate. `WeaSEL` can then train end model f to generalize the relationship between these labels and the features of the data. On the other hand, assume end

model f already provides accurate predictions for our data. These predictions can thus be used as supervision to train g to output correct probabilistic labels. The `stop-grad` operation naturally encodes this interpretation *i.e.* each model treats the other's prediction as the target.

In this work, we train WeaSEL in a federated setting, where the objectives become

$$L_F = \mathbb{E}_{\mathcal{D}^k \sim \mathcal{P}}[L_f(\mathcal{D}^k)] \qquad \text{and} \qquad L_G = \mathbb{E}_{\mathcal{D}^k \sim \mathcal{P}}[L_g(\mathcal{D}^k)].$$

Because we have access to a finite number of clients, and a finite sample of examples per client, we use empirical risk minimization to solve for these objectives.

We exchange f, g and $\mathcal{L}^*$ throughout training. We assume global class balances $P(y)$ to be known, as is frequent in related work (Boecking et al., 2020; Ratner et al., 2019; Fu et al., 2020; Chen et al., 2021). Other works in centralized settings have proposed ways of estimating this quantity from validation data or from LF responses (Ratner et al., 2019). We leave the problem of estimating $P(y)$ from federated data as a direction of future work, and explore the interaction between a global class balance $P(y)$ and local client balances $P_k(y)$ in Section 5.5.2.

Assumptions

WSHFL relies on the ability to generate candidate LFs of varying quality, for which we use domain-specific processes. Previous work in mining LFs has observed that this generation process is possible for several applications (Varma & Ré, 2018; Boecking et al., 2020). We also rely on the ability of experts to determine whether a given LF is accurate. Once again, prior work has shown that domain experts are able to exercise this judgment, either while providing feedback of this type (Boecking et al., 2020), or while crafting LFs from scratch (Goswami et al., 2021; Dey et al., 2022; Fries et al., 2019; Dunnmon et al., 2020).

In this work we assume that the parameterized LFs can be freely shared with the server and, after aggregation and inspection by the expert, with other clients. We also assume estimates $\hat{u}_j, j \in A$, for a given A to be independent in order to construct our lower bound on the posterior $P(u' = 1|\lambda', Q)$. This independence will not hold, for example, if the distribution over clients $\mathcal{P}$ changes over time (Kairouz et al., 2021).

5.3 Labeling Function Generation

We discuss two data modalities: text and time-series. For each modality, we parameterize LFs differently and use different strategies to generate and aggregate them.

5.3.1 Text LFs

For text data, we propose LFs parameterized by a unigram and label, assigning the label if the unigram is present in the data point (see Figure 5.2). Otherwise, the LF abstains. Previous studies applying programmatic weak supervision to text data have found n-grams to be excellent sources of supervision (Gao et al., 2022; Boecking et al., 2020). This is particularly true for unigrams due to their larger support in data.

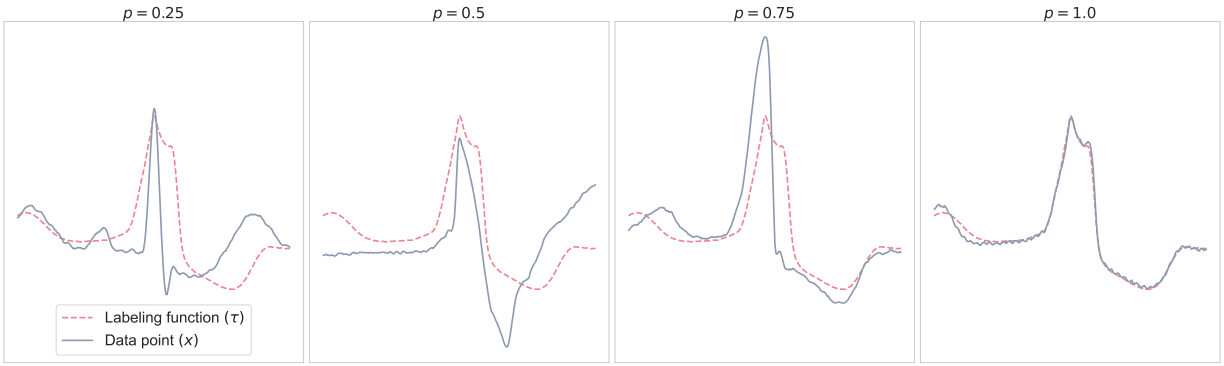

Figure 5.3: Example of a time-series labeling function representing an arrhythmia. We show 4 data points with increasing probabilities of belonging to the given class. These examples will be labeled as arrhythmias as we vary our probability threshold p.

Given this parameterization, a client can automatically generate $\mathcal{L}_k$ from the cross product of the set of possible labels and the unigrams in its vocabulary within a document frequency range[1]. Meanwhile, in the server, we can aggregate candidates with the same unigram and label.

5.3.2 Time-series LFs

For time-series data, our LFs are fully-parameterized using a three-tuple $(\tau,\ d_\tau,\ l)$, where $\tau \in \mathbb{R}^d$ is a time-series template, $d_\tau \in \mathbb{R}$ is a distance threshold, and $l \in \mathcal{Y}$ denotes the label. Given these parameters, a distance function $\mathbf{d} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, and a probability threshold p, each time-series LF has the following functional form:

$$\lambda(x; \tau, d_\tau, l) = \begin{cases} l, & \mathcal{F}(d_\tau, \mathbf{d}(x, \tau)) \geq p \\ 0, & \text{otherwise} \end{cases}, \text{ where } \mathcal{F}(x, x_0) = \frac{1}{1 + \exp -\{x - x_0\}}.$$

Intuitively, the more a time-series x looks like template τ, the higher the probability it has of being assigned label l. p is a fixed user-defined parameter which transforms distance into a more interpretable notion of probability. We present an example of our time-series LFs in Figure 5.3. In this study, we use normalized euclidean distance as the distance function $\mathbf{d}$, since it is widely used in time-series data mining, can be easily computed in $\mathcal{O}(d)$, and has been empirically shown to be competitive with other complex distance measures across several domains (Ding et al., 2008; Mueen et al., 2009).

To generate $\mathcal{L}_k$, each client can use the k-means algorithm with normalized euclidean distance to find m clusters in its data. Each of the m centroids can then be a template for m different labeling functions. We define d_τ to be the distance of the cluster member farthest from the centroid (template). Finally, each client can finish constructing $\mathcal{L}_k$ by taking the cross-product of the set of possible labels and these templates.

[1]The document frequency of a unigram is defined as the fraction of documents which contain at least one occurrence of the unigram.

To aggregate these candidates on the server, we cluster LFs from multiple clients. Specifically, we cluster templates of LFs with the same label using the k-means clustering algorithm. Each cluster then represents an aggregate LF: the cluster centroid serves as the template τ, and the maximum d_τ of any cluster member serves as the distance threshold of the aggregate.

5.4 Experimental Setup

Datasets. We demonstrate our method on two data modalities: text and time-series data. In Appendix D.1, we provide further details about our datasets. In our experiments, we consider binary classification tasks, but WSHFL is also formulated to solve multi-class problems.

For text, we use two datasets frequently used to benchmark text classification models: the Amazon product reviews dataset (Ni et al., 2019) and the IMDb movie reviews dataset (Maas et al., 2011). In both of these datasets we solve a binary sentiment analysis task. On the Amazon dataset, we treat each unique reviewer as a different client, whereas on the IMDb dataset, we split reviews uniformly at random between clients.

For time-series, we use the Massachusetts Institute of Technology – Beth Israel Hospital Arrhythmia Database (MIT BIH) dataset (Moody & Mark, 2001; Goldberger et al., 2000). It consists of 48 half-hour excerpts from 47 subjects and contains beat-level annotations for a wide range of heart beats. We solve a binary classification task of discriminating normal heart beats from arrhythmias and treat each patient as a different client.

Methods and Models. We featurize our text data using a pre-trained open-source sentence transformer (Reimers & Gurevych, 2019). Meanwhile, for our arrhythmia detection task, we use the raw ECG data[2] (sampled at 360Hz) and output a prediction for each window of 256 samples around given peaks. Client models h_k and label model g are always two-layer perceptrons. For our text datasets, the end model f is also a two-layer perceptron, while for our time-series data we use a one-dimensional convolutional neural network. We optimize our parameters on our validation dataset, and report the area under the receiver operating characteristics curve (ROC AUC) on the test dataset. We provide further details about our models and hyper-parameters in Appendix D.1 and Appendix D.2.

Expert. As an expert, we use an oracle that labels a LF as useful if it has an accuracy in the training data of at least 0.7. We perform experiments with different thresholds in Appendix D.6.

Baselines. We compare the predictive performance of WSHFL using $\delta = 0.05$ against three baselines:

1. Random: We show random aggregated LFs to the expert. This corresponds to changing line 14 in Algorithm 1 to $\lambda \leftarrow \texttt{SelectRandom}(\mathcal{L}')$.

[2]We only use the Modified Limb lead II (MLII) obtained by placing electrodes on the chest, as is done in prior work (Goswami et al., 2021).

2. Naive Greedy: At each time-step t, we show the expert the $\lambda' \in \mathcal{L}'$ with the highest

$$P(u' = 1|\lambda', Q_{t-1}) = \frac{1}{|A|} \sum_{j \in A} \hat{u}_j.$$

   Simple mean estimates are natural in federated settings, but they may have high variance in this particular scenario.
3. Supervised: We also present a baseline that uses all ground truth labels and is trained using FedAvg (McMahan et al., 2017).

5.5 Results and Discussion

In this section, we investigate the following three hypothesis:

1. We can mine LFs with desireable properties in a federated setting.
2. Given good LFs and unlabeled distributed data, we can learn a performant federated model.
3. We can combine both of these steps in WSHFL, i.e., mining good LFs and training a performant end model using on-device data.

$u_j = 1$		
	Percentage	**Coverage**
	Amazon	
WSHFL	**30.13 +/- 5.27 %**	**1.94 +/- 2.28 %**
Greedy	20.93 +/- 4.70 %	0.04 +/- 0.04 %
Random	9.20 +/- 3.55 %	0.03 +/- 0.06 %
	IMDb	
WSHFL	34.13 +/- 8.33 %	**3.47 +/- 3.40 %**
Greedy	**43.07 +/- 3.63 %**	0.03 +/- 0.03 %
Random	21.20 +/- 3.55 %	0.07% +/- 0.37 %
	MIT BIH	
WSHFL	**97.80 +/- 1.40 %**	2.41 +/- 3.87 %
Greedy	96.40 +/- 2.50 %	**3.83 +/- 6.46 %**
Random	44.20 +/- 6.60 %	0.63 +/- 2.47 %

Table 5.1: Percentage of LFs labeled as $u_j = 1$ out of those inspected by the expert, and their mean coverage. We can see how WSHFL mines both high accuracy and high coverage LFs for all of our datasets. We present the mean and standard deviation across five repetitions. In bold, the highest mean per dataset.

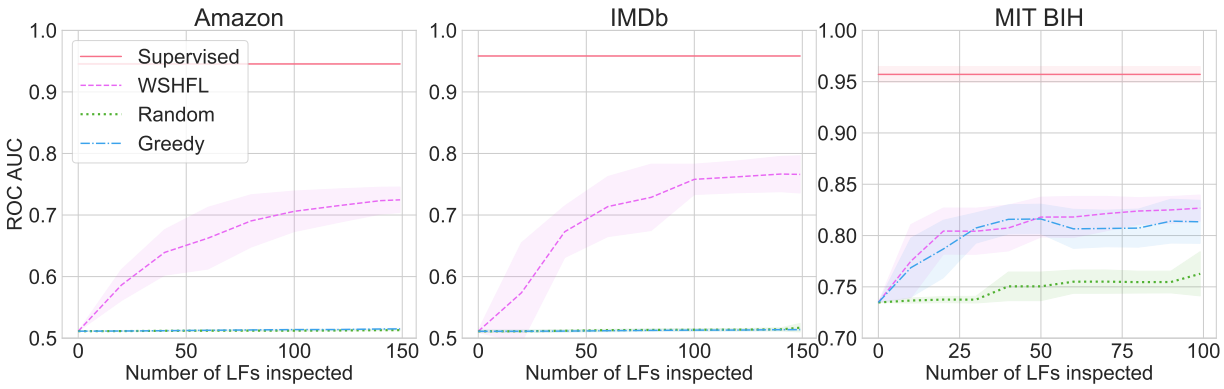

Figure 5.4: Results for a majority vote classifier given mined LFs. We observe how, as we present more LFs to the expert, WSHFL outperforms our baselines on our text datasets, and performs comparably to greedy on MIT BIH. Time-step 0 corresponds to an initialization as described in Appendix D.4. We repeat each experiment five times with different random seeds and show the mean (line) and standard deviation (shaded).

5.5.1 Automatic Mining of LFs

Previous work on automatic mining of LFs has shown the importance of obtaining candidates with both high coverage and a high accuracy gap above chance (Boecking et al., 2020), where the coverage $l_j = P(\lambda_j(x) \neq 0)$ is the frequency at which λ_j does not abstain. We plot these two quantities for the LFs inspected by the expert in Figure 5.8. Likewise, in Table 5.1, we present the percentage of LFs labeled as $u_j = 1$ out of those inspected, and their mean coverage. We observe how WSHFL promotes the mining of both high accuracy and high coverage heuristics across data modalities. Meanwhile, our greedy baseline fails to find high coverage LFs for our text dataset, successfully mining high coverage LFs only in our time-series experiments with the MIT BIH dataset.

To understand this behaviour, in Appendix D.7 we sketch the distribution of the proposed candidates' accuracies and coverages for our datasets. We observe how, for Amazon and IMDb, high accuracy candidates tend to have low coverage. Thus, if our posterior estimates are correct, the naive greedy baseline will end up with low coverage LFs. However, this is not the case for MIT BIH, where candidates with high accuracy also have good coverage. Thus, we expect greedy to be a competitive baseline for this dataset.

In Figure 5.4, we also test the mined LFs on a simple downstream task: a centralized majority vote classifier, which is a competitive baseline in the PWS literature (Rühling Cachay et al., 2021; Gao et al., 2022; Dey et al., 2022). We see how only WSHFL shows any meaningful improvement for our text datasets (Amazon and IMDb) while it performs comparably to our greedy baseline on the MIT BIH dataset.

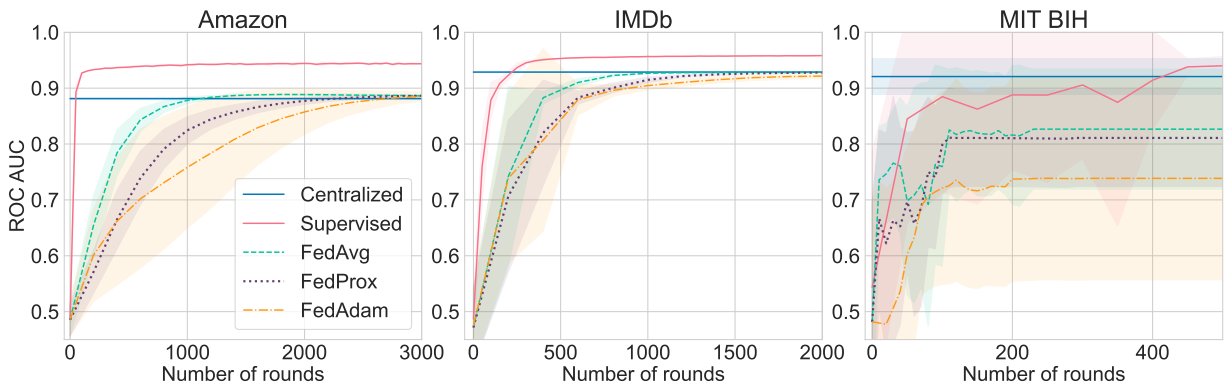

Figure 5.5: Results of training WeaSEL in a federated setting given a set of curated LFs. We observe that, given enough rounds of communication, we can match or come close to the performance of a centralized training scheme. We repeat each experiment five times with different random seeds.

5.5.2 Training of the PWS Model

We validate that we can successfully train a WeaSEL model as proposed by Rühling Cachay et al. (2021) (Section 5.2.3) in a federated manner. For these experiments, we use a pre-curated set of LFs which we describe in Appendix D.3. In Figure 5.5, we show the results of our experiments. We present two baselines: the fully supervised baseline described above and an additional baseline corresponding to training WeaSEL in a centralized manner. With the latter, we aim to corroborate the utility of the used LFs in a previously studied setting of reduced complexity.

We study the behavior of federated training with three different algorithms: FedAvg, FedProx, and FedAdam. We observe how, for Amazon and IMDb, all three algorithms match the centralized performance after sufficient number of communication rounds. For the MIT BIH dataset, the best performing algorithm (FedAvg) achieves an ROC AUC of 82.66% vs. 92.08% of the centralized performance.

We also explore the effects of class imbalance on the performance of federated training. We use the method proposed by Hsu et al. (2019), which parameterizes the class distribution on a client by a vector $\mathbf{q} \sim \text{Dir}(\alpha \mathbf{p})$, where $\mathbf{p}$ is a uniform prior and $\alpha > 0$ controls how much the class distributions across clients resemble each other. In Figure 5.6, we show results for training WeaSEL using FedAvg on the IMDb dataset. We observe how our results are consistent as we vary α: when clients have identical class distributions ($\alpha \to \infty$), when clients have only one class each ($\alpha \to 0$), and for intermediate values. These results also suggest that it may be sufficient to specify the global class balance $P(y)$ even when the clients' $P_k(y)$ differ.

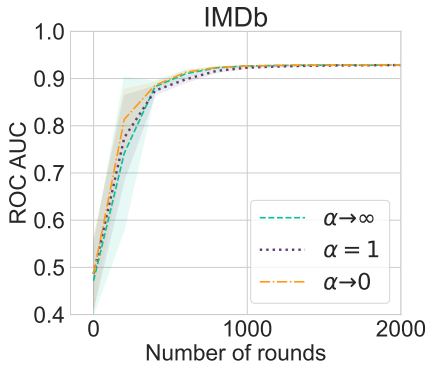

Figure 5.6: Results of training WeaSEL in a federated setting on the IMDb dataset, given a set of curated LFs. We achieve consistent results as we vary the class distributions in each client, with $\alpha \to \infty$ corresponding to balanced classes, and $\alpha \to 0$ corresponds to one class per client. We present the test ROC AUC of the end model vs. the number of rounds of federated training. We repeat each experiment five times with different random seeds.

5.5.3 Putting It All Together

Finally, we demonstrate that we both mine the LFs and use these LFs to train a federated model using FedAvg. We show our results in Figure 5.7. In our text datasets (Amazon and IMDb), we see how WSHFL is both more effective and efficient than our baselines at leveraging the expert's supervision. Meanwhile, for the MIT BIH dataset, it performs comparably to our greedy baseline.

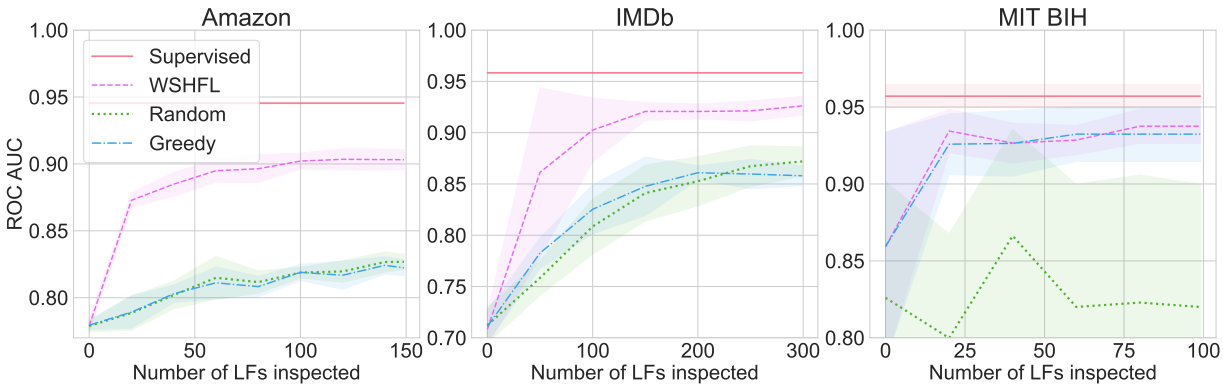

Figure 5.7: Results for WSHFL on our datasets. We observe how, as we present more LFs to the expert, WSHFL outperforms our baselines on our text datasets, and performs comparably to greedy on MIT BIH. Time-step 0 corresponds to an initialization as described in Appendix D.4. We repeat each experiment five times with different random seeds and show the mean (line) and standard deviation (shaded).

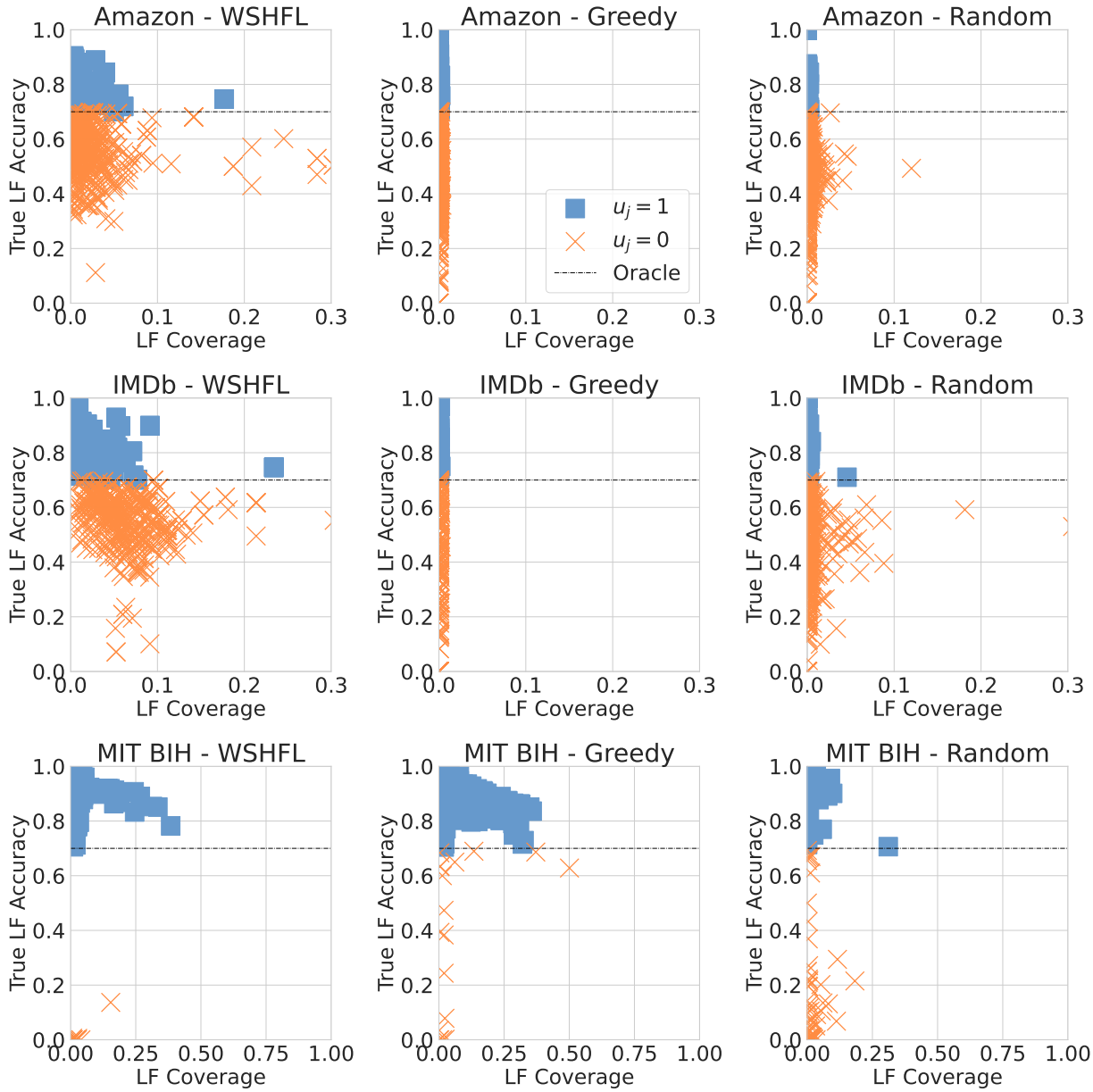

Figure 5.8: Training accuracies vs. coverages for all LFs inspected by the expert across five repetitions. We qualitatively observe how WSHFL promotes high accuracy and high coverage LFs across our three datasets. The black dotted line is the threshold at which our oracle starts labeling $u_j = 1$ in a LF, which we set to 0.7.

5.5.4 Societal Impact and Future Work

Societal Impact. As with most federated learning algorithms, WSHFL makes frequent exchanges between clients and a central server. These exchanges come in the form of parameterized models and LFs. Through these parameters, some information may leak from the clients. Understanding and mitigating the harms of exchanging model parameters is an active area of study (McMahan & Ramage, 2017; Li et al., 2019a; Bonawitz et al., 2022). However, these harms may not be fully understood depending on the LF parameterization, e.g., unigrams extracted from client data may leak private information or contain terms that the expert deems offensive. Future work must understand the potential risks and harms of sharing a particular type of LF, and develop mechanisms to mitigate these harms.

Studies in specific application domains. To further establish the utility of the proposed approach, future work should study its performance and viability in applications and modalities beyond those explored in this chapter, e.g., clinical tabular or image data. A salient challenge in these studies will be the definition of LFs that can be easily inspected by experts. This research direction will also benefit from conducting user studies to evaluate proposed LF generation mechanisms (Boecking et al., 2020). Immediate future work could conduct a study with clinical experts to evaluate our proposed time-series LFs.

Improve selection of LFs. We validate that WSHFL's selection strategy finds both accurate and high coverage LFs. However, there are ways in which this selection strategy could improve. For example, future work could extend the active search formulation presented in this chapter using non-myopic strategies (Jiang et al., 2017) in the federated setting. Future studies could also equip the selection strategy with exploration capabilities, as is common in other sequential decision making settings (Sutton & Barto, 2018). Finally, we could conceive an active search formulation that is aligned with the performance of the end model itself, or with other properties of LFs, e.g., LF overlaps.

Part IV

Open-Source Contributions

Modern networks of wearable devices, mobile phones, or autonomous vehicles, generate massive amounts of data each day. This wealth of data can help to learn models that can improve the user experience on each device. However, the scale and heterogeneity of this data presents new challenges in research areas such as collaborative learning, federated learning, meta-learning, and multi-task learning. As the machine learning community tackles these challenges, we need to ensure that developments made in these areas are grounded with realistic benchmarks. To this end, we propose LEAF, a modular benchmarking framework for learning in cross-device settings. LEAF includes a suite of open-source datasets, a rigorous evaluation framework, and a set of reference implementations, all geared towards capturing the obstacles and intricacies of practical mobile device environments.

Chapter 6

LEAF: A Benchmark for Cross-Device Settings

With data increasingly being generated on networks of remote devices, there is growing interest in empowering on-device applications with models that make use of such data (McMahan et al., 2017; Smith et al., 2017; Kairouz et al., 2021; Wang et al., 2021). Learning on data generated in these networks, however, introduces several new obstacles:

Statistical: Data is generated on each device in a heterogeneous manner, with each device associated with a different, though perhaps related, underlying data generating distribution. Moreover, the number of data points typically varies significantly across devices.

Systems: The number of devices in these networks is typically order of magnitudes larger than the number of nodes in a typical distributed setting, such as datacenter computing. In addition, each device may have significant constraints in terms of storage, computational, and communication capacities. Furthermore, these capacities may also differ across devices due to variability in hardware, network bandwidth, and power. Thus, these settings may suffer from communication bottlenecks that dwarf those encountered in traditional datacenter settings, and may require faster on-device inference.

Privacy and Security: Finally, the sensitive nature of personally-generated data requires methods that operate on on-device data to balance privacy and security concerns with more traditional considerations such as statistical accuracy, scalability, and efficiency (McMahan et al., 2018; Li et al., 2019a; Bonawitz et al., 2022).

Previous works propose ways of dealing with these challenges and train collaborative models from this data. However, they fall short when it comes to their experimental evaluation. As an example, consider the federated learning paradigm, which focuses on training collaborative models directly on the devices' networks (McMahan et al., 2017; Kairouz et al., 2021). Experimental works focused on federated learning broadly utilize three types of datasets, each with their own shortcoming: (1) datasets that are commonly used and yet do not provide a realistic model of a federated scenario, e.g., artificial partitions of MNIST, fashion-MNIST or CIFAR-10 (Konečný et al., 2016; McMahan et al., 2017; Geyer et al., 2017; Bagdasaryan et al., 2020; Kamp et al., 2019; Ulm et al., 2018; Wang et al., 2019); (2) realistic but proprietary federated datasets, e.g., data from an unnamed social network in (McMahan et al., 2017), crowdsourced

voice commands in (Leroy et al., 2019), and proprietary data by Huawei in (Chen et al., 2018); and (3) realistic federated datasets that are derived from publicly available data, but which are not straightforward to reproduce, e.g., FaceScrub in Melis et al. (2018), Shakespeare in McMahan et al. (2017) and Reddit in Konečný et al. (2016); McMahan et al. (2018); Bagdasaryan et al. (2020).

As a second example, consider meta-learning, a related learning paradigm proposed by Chen et al. (2018); Khodak et al. (2019b) as a way to tackle the statistical challenges of cross-device networks. The paradigm is indeed a natural fit for these settings, as the heterogeneous devices can be interpreted as meta-learning tasks. However, popular meta-learning benchmarks such as *Omniglot* (Lake et al., 2011; Finn et al., 2017; Vinyals et al., 2016; Snell et al., 2017) and *miniImageNet* (Ravi & Larochelle, 2016; Finn et al., 2017; Vinyals et al., 2016; Snell et al., 2017) focus on k-shot learning where all tasks have the same number of samples and each class has the same number of samples in each task, failing to capture the real-world challenges that on-device data would bring to meta-learning solutions. In fact, all of the previously mentioned datasets could thus be categorized as the first type mentioned above: popular yet unrealistic for our purposes.

As a final example, consider multi-task learning (MTL). This paradigm is also amenable to cross-device settings (Smith et al., 2017) but, contrary to realistic networks of devices, it is usually explored in regimes with small numbers of tasks and samples, e.g., the popular *Landmine Detection* (Zhang & Schneider, 2010; Murugesan & Carbonell, 2017; Xue et al., 2007; Smith et al., 2017), *Computer Survey* (Argyriou et al., 2008; Agarwal et al., 2010; Kumar & Daumé III, 2012) and *Inner London Education Authority School* (Murugesan & Carbonell, 2017; Lee et al., 2016; Agarwal et al., 2010; Argyriou et al., 2008; Kumar & Daumé III, 2012) datasets have at most 200 tasks each. We highlight that, while federated learning, meta-learning, and multi-task learning are the presented applications for LEAF, the framework in fact encompasses a wide range of potential learning settings, such as collaborative learning, on-device learning or inference, transfer learning, life-long learning, and the development of personalized learning models.

Our work aims to bridge the gap between artificial datasets that are popular and accessible for benchmarking, and those that realistically capture the characteristics of a network of devices. Moreover, beyond establishing a suite of cross-device datasets, we propose a clear methodology for evaluating methods and reproducing results. To this end, we present LEAF, a modular benchmarking framework geared towards learning in massively distributed networks of remote devices.

6.1 LEAF

LEAF is an open-source benchmark for cross-device settings. [1] It consists of (1) a suite of open-source datasets, (2) an array of statistical and systems metrics, and (3) a set of reference implementations. As shown in Figure 6.1, LEAF's *modular* design allows these three components to be easily incorporated into diverse experimental pipelines. We proceed to detail LEAF's core components.

Datasets: We have curated a suite of realistic cross-device datasets for LEAF. We focus on

[1] All code and documentation can be found at https://github.com/TalwalkarLab/leaf/.

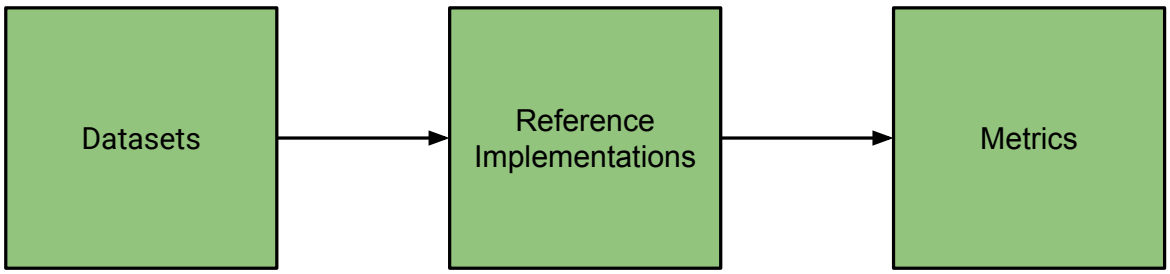

Figure 6.1: LEAF modules. The "Datasets" module preprocesses the data and transforms it into a standardized format, which can integrate into an arbitrary ML pipeline. LEAF's "Reference Implementations" module is a growing repository of common collaborative methods used in cross-device settings, with each implementation producing a log of various different statistical and systems metrics. Any log generated in an appropriate format can then be used to aggregate and analyze these metrics in various ways through LEAF's "Metrics" module.

datasets where (1) the data has a natural keyed generation process (where each key refers to a particular device/user); (2) the data is generated from networks of thousands to millions of devices; and (3) the number of data points is skewed across devices. Currently, LEAF consists of six datasets:

- *Federated Extended MNIST (FEMNIST)*, which is built by partitioning the data in Extended MNIST (LeCun, 1998; Cohen et al., 2017) based on the writer of the digit/character.
- *Sentiment140* (Go et al., 2009), an automatically generated sentiment analysis dataset that annotates tweets based on the emoticons present in them. Each device is a different twitter user.
- *Shakespeare*, a dataset built from *The Complete Works of William Shakespeare* (William Shakespeare. The Complete Works of William Shakespeare, n.d.; McMahan et al., 2017). Here, each speaking role in each play is considered a different device.
- *CelebA*, which partitions the Large-scale CelebFaces Attributes Dataset [2] (Liu et al., 2015) by the celebrity on the picture.
- *Reddit*, where we preprocess comments posted on the social network on December 2017.
- A *Synthetic* dataset, which modifies the synthetic dataset presented in Li et al. (2020c) to make it more challenging for current meta-learning methods. See Appendix E.1 for details.

We provide statistics on these datasets (except the Synthetic one, as these vary depending on the user's settings) in Table 6.1. In LEAF, we provide all necessary pre-processing scripts for each dataset, as well as small/full versions for prototyping and final testing. Moving forward, we plan to add datasets from different domains (e.g. audio, video) and to increase the range of machine learning tasks (e.g. text to speech, translation, compression, etc.).

[2] The original CelebA data is hosted in http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html

Dataset	Number of devices	Total samples	Samples per device	
			mean	stdev
FEMNIST	3,550	805,263	226.83	88.94
Sent140	660,120	1,600,498	2.42	4.71
Shakespeare	1,129	4,226,158	3,743.28	6,212.26
CelebA	9,343	200,288	21.44	7.63
Reddit	1,660,820	56,587,343	34.07	62.95

Table 6.1: Statistics of datasets in LEAF.

Metrics: Rigorous evaluation metrics are required to appropriately assess how a learning solution behaves in cross-device scenarios. Currently, LEAF establishes an initial set of metrics chosen specifically for this purpose. For example, we introduce metrics that better capture the entire distribution of performance across devices: performance at the 10th, 50th and 90th percentiles and performance stratified by natural hierarchies in the data (e.g. "play" in the case of the Shakespeare dataset or "subreddit" for Reddit). We also introduce metrics that account for the amount of computing resources needed from the edge devices in terms of number of FLOPS and number of bytes downloaded/uploaded. Finally, LEAF also recognizes the importance of specifying how the accuracy is weighted across devices, e.g., whether every device is equally important, or every data point equally important (implying that power users/devices get preferential treatment).

Reference implementations: In order to facilitate reproducibility, LEAF also contains a set of reference implementations of algorithms geared towards cross-device scenarios. Currently, this set is limited to the federated learning paradigm, and in particular includes reference implementations of minibatch SGD and FedAvg (McMahan et al., 2017).

6.2 LEAF in action

We now show a glimpse of LEAF in action. In particular, we highlight three of LEAF's characteristics[3]:

LEAF enables reproducible science: To demonstrate the reproducibility enabled via LEAF, we focus on qualitatively reproducing the results that (McMahan et al., 2017) obtained on the Shakespeare dataset for a next character prediction task. In particular, it was noted that for this particular dataset, the FedAvg method *diverges* as the number of local epochs increases. Results are shown in Figure 6.2, where we indeed see similar divergence behavior in terms of the training loss as we increase the number of epochs.

LEAF provides granular metrics: As illustrated in Figure 6.3, our proposed systems and statistical metrics are important to consider when serving multiple clients simultaneously. For statistical metrics, we show the effect of varying the minimum number of samples per user in

[3]For experiment details, see Appendix E.2.

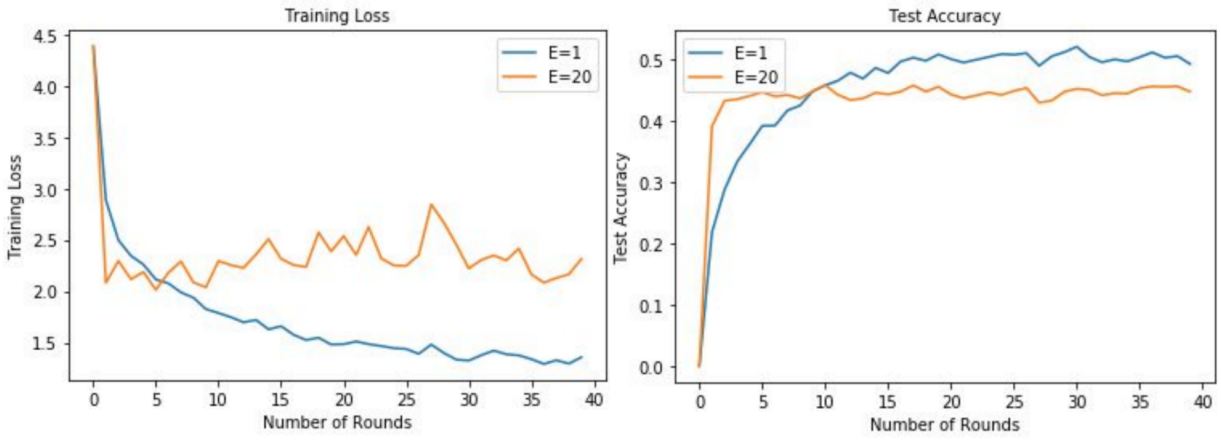

Figure 6.2: Convergence behavior of FedAvg on a subsample of the Shakespeare dataset. We are able to achieve a per sample test accuracy comparable to the results obtained in McMahan et al. (2017). We also qualitatively replicate the divergence in training loss that is observed for large numbers of local epochs (E).

Sentiment140 (which we denote as k). We see that, while median performance degrades only slightly with data-deficient users (i.e., $k = 3$), the 25th percentile degrades dramatically. Meanwhile, for systems metrics, we run minibatch SGD and FedAvg for FEMNIST and calculate the systems budget needed to reach a per sample accuracy threshold of 0.75. We characterize the budget in terms of total number of FLOPS across all devices and total number of bytes uploaded to network. Our results demonstrate the improved systems profile of FedAvg when it comes to the communication vs. local computation trade-off, though we note that in general methods may vary across these two dimensions.

LEAF is modular: To demonstrate LEAF's modularity, we incorporate its "Datasets" module into three new experimental pipelines: one that trains purely local models for each device (on CelebA and our Synthetic dataset), one that disregards the natural partition between devices, i.e., it mixes all the data (on Reddit), and one in which we use the popular meta-learning method *Reptile* (Nichol et al., 2018) (on FEMNIST). Results for these experiments are presented in Table 6.2. These particular pipelines shed light on how different modeling approaches may be more or less appropriate for different datasets (Jiang et al., 2019; Khodak et al., 2019b).

6.3 Conclusions and Impact

We present LEAF, a modular framework for learning in ecosystems marked by massively distributed networks of devices. Learning paradigms applicable in such settings include collaborative learning, federated learning, meta-learning and multi-task learning. LEAF allows researchers and practitioners in these domains to reason about new proposed solutions under realistic assumptions of on-device data.

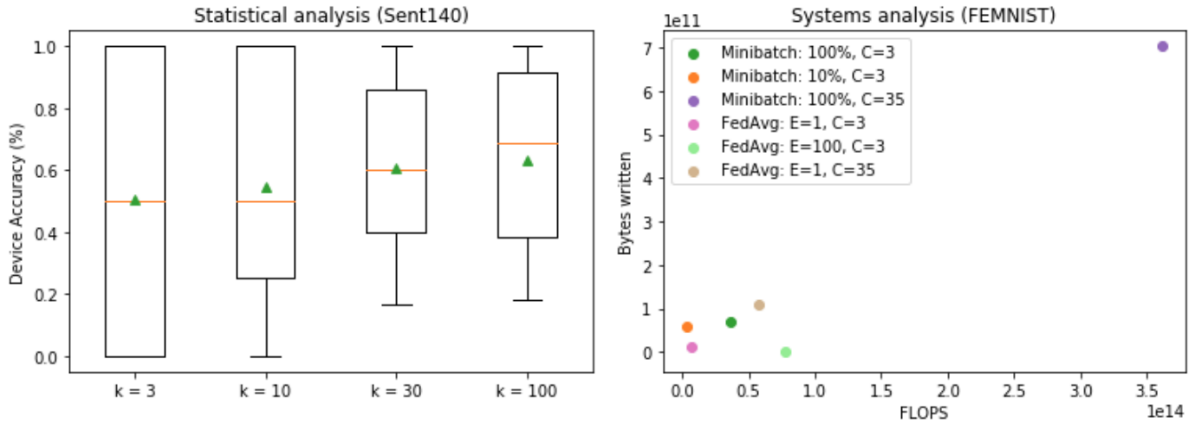

Figure 6.3: Statistical and Systems analyses for Sent140 and FEMNIST. For Sent140: k is the minimum number of samples per user. Orange lines represent the median device accuracy, green triangles represent the mean, boxes cover the 25th and 75th percentile, and whiskers cover the 10th to the 90th percentile. For FEMNIST: C is the number of clients selected per round, and E is the number of epochs each client trained locally for FedAvg. For minibatch SGD we report the percentage of data used per client.

Dataset	**FedAvg (baseline)**	**Additional Pipeline**	
		description	accuracy
CelebA	89.46%	Local Models	65.29%
Synthetic	71.89%		87.34%
Reddit	13.35%	Global IID model	12.60%
FEMNIST	74.72%	Reptile	80.24%

Table 6.2: Demonstration of LEAF's modularity. We incorporate LEAF's datasets into new experimental pipelines (beyond FedAvg) and report the resulting sample test accuracies.

Since its inception, LEAF has had a wide impact in the federated learning community. Its open-source implementations have facilitated reproducible research (Laguel et al., 2021; Pillutla et al., 2022) and its datasets have become a reference for cross-device federated learning (Ogier du Terrail et al., 2022). In particular, other popular federated learning benchmarks and frameworks have adopted LEAF's datasets for experimentation. For example, Tensorflow Federated (TFF) (Bonawitz et al., 2019) hosts the FEMNIST, Shakespeare and CelebA datasets. Meanwhile, FLSim (Meta AI Research, 2022) and Flower (Beutel et al., 2022) define explicit ports to conduct studies with LEAF datasets.

We can also quantify LEAF's impact through the usage of its github repository, which has over 200 forks and 700 stars at the time of writing. However, these statistics probably underestimate its actual usage, as some of LEAF's more popular datasets are hosted by TFF.

Moving forward, we hope to keep updating LEAF to reflect the realities and findings of the research community. For example, we want to add:

- Reference implementations in different machine learning frameworks, e.g., PyTorch (Paszke et al., 2019) and JAX (Bradbury et al., 2018).
- Principled ways to vary the heterogeneity of the datasets (Hsu et al., 2019).
- Datasets, implementations and metrics for semi-supervised tasks (Jeong et al., 2020; Caldas et al., 2023) and broader collaborative learning tasks Mittal et al. (2022).

Our hope is to expand the number of scenarios in which LEAF can facilitate reproducible and grounded research, contributing to the healthy development of the fields of collaborative and federated learning.

Conclusions

Chapter 7

Conclusions

This dissertation studies how to improve the practical utility of collaborative learning systems that leverage siloed data. We do this by studying requirements imposed on these systems by applications of real-world interest. In particular, we focus on three settings:

- Collaborative models learned on heterogeneous networks, where communication constraints are a crucial bottleneck.
- Healthcare models learned across clinical collaborators, where explanations are an important component of clinical utility.
- Collaborative systems that use unlabeled on-device data, where encoding expert knowledge is an ongoing challenge.

In each of these settings we propose methods and algorithms to tackle each of the identified requirements. Furthermore, we examine the interplay of the studied dimension with predictive performance, as it continues to be a critical functional attribute by which learning systems are evaluated. Lastly, we discuss the impact of our work and future directions that study the relation between each dimension and other properties of collaborative learning systems, e.g., privacy and fairness. As particular motivators and case studies, we used clinical and on-device data, but the proposed techniques can be extended to different application domains.

This dissertation also introduces a benchmark for collaborative learning in cross-device settings, e.g., when the collaborators are mobile devices. The benchmark addresses shortcomings of previous experimental evaluations in the field, particularly in regard to the datasets used. We examine the impact of this contribution in the field of federated learning, and discuss possible extensions that would expand its reach.

One of the main themes of this dissertation is collaboration. In particular, collaboration among parties that cannot share data and must rely on other forms of information exchange. In each part, we rely on a different type of information exchange: gradients in Part I, fully trained models in Part II, and labeling functions in Part III. To turn this information into actionable knowledge, we train a collaborative model to make predictions. Still, this knowledge could be exploited even further by keeping track of what is learned, when is it learned, and by whom. This type of management requires both explicit tracking tools and a deeper understanding of dynamic collaborative systems, e.g., where the client distribution changes over time.

Another central theme of this thesis is utility, in the sense of training models that will be truly used in practice. For a given setting, this implies understanding the constraints of the model and the users that will interact with it. We wish to highlight the latter: the users interacting with the collaborative system. Throughout this dissertation, we progressively mature our tools to capture our information about users. In Part II we motivate our work on explanations based on clinician feedback. Meanwhile, in Part III we use tools from sequential decision to explicitly build a model of the expert's beliefs. A later stage of maturity would draw tools from the social sciences in order to design better user models, validate their assumptions and conduct user studies.

Adoption of Collaborative Learning Systems

This dissertation focuses on two particular types of collaboration: between mobile devices and among clinical centers. Throughout the thesis, we have highlighted differences and similarities between collaborative learning in the two scenarios, e.g., formalizing them as cross-device and cross-silo federated learning, respectively. Now, we wish to further contrast these settings along a new axis: the barriers to adoption of these collaborative systems.

On one hand, learning collaborative models from on-device data is gaining traction in practice with products such as Google's Gboard and Google Assistant already deploying federated learning solutions (McMahan & Thakurta, 2022). This adoption has been facilitated by factors characteristic of this ecosystem, including:

- A central entity (e.g., Google, Apple) that bears the burden of development.
- Clear incentives for adoption, e.g., data privacy and generating user trust.
- An existing network over which devices can communicate with a central server.

On the other hand, adoption of collaborative learning among clinical collaborators can prove more challenging, as it is an ecosystem marked by:

- Multiple entities sharing the responsability of development, i.e., the centers.
- The possibility of misaligned incentives, as some centers may benefit differently from the collaboration.
- The absence of a common network over which centers can communicate.

These issues manifest into additional practical considerations that may not be the focus of current research in collaborative learning. For example, research into scalable methods for data harmonization would remove a major bottleneck for multi-center studies (Nan et al., 2022; Williams et al., 2023), who need to unify data models and distributions among collaborators. Previous work has shown that harmonization leads to better performing collaborative models (Shukla et al., 2018; Raghu et al., 2019; Marzi et al., 2022), yet the sources of disharmony are multiple and diverse, e.g., device and laboratory variability.

Ultimately, however, the adoption of collaborative learning methods in clinical settings hinges on showing that they lead to actual improvements in patient outcomes. Thus, we believe that current studies should focus on demonstrating these meaningful improvements for concrete applications, as these success stories can later materialize into concrete incentives to overcome the mentioned bottlenecks.

Future Work and Open Questions

This thesis used the framework of federated learning to contextualize its contributions. Still, there are other recent collaborative learning frameworks studied by the machine learning community (Raffel, 2023; Blum et al., 2017). In particular, we wish to highlight the study of modular architectures (Mittal et al., 2022; Rosenbaum et al., 2019) as an avenue to allow stakeholders to train specialized modules, leading to more maintainable and transparent collaborative systems. This would be useful in healthcare settings, where probing the model has been shown to be of special interest.

It is also timely to study the relationship between collaborative learning and large language models. In one direction, researchers can study how to use these pre-trained models to facilitate collaborative learning, using them to improve the utility trade-offs of cross-device federated learning (Wang et al., 2023), or leveraging in-context learning (Xie et al., 2021) to produce labeling functions for unlabeled, siloed data. In the opposite direction, we can study how to use collaborative learning to train these large models on user compute (Borzunov et al., 2022; Yuan et al., 2022) in order to allow new stakeholders to train these complex models. This relates to our previous ideas on modularity, as we may need to understand how to distribute the different components of these models, e.g., how to generate sub-models for these new architectures. The community will also need to study users' incentives to share data and compute, and create practical mechanisms to account for these.

We outline more specific future work in each of our chapters. An underlying theme of the proposed directions is to research the interplay of the different dimensions of collaborative systems, outside of predictive performance, e.g., understand how differential privacy can affect the explanations presented in Part II. Another recurrent theme is understanding and mitigating the harms of exchanging information among collaborators. Siloing the data follows the principles of focused data collection and minimization (Kairouz et al., 2021), but the parameter exchanges we work with in this thesis still leak information about the stakeholders.

Finally, we want to mention the need to study ways to regulate collaborative models. Going forward, the community will have to think carefully about questions such as: how to build test sets if the data is distributed, sensitive and unlabeled? how to ensure stakeholders are properly compensated if their data is used? how to compensate them if their compute is used? how can they withdraw consent from participation? Timely work in this direction will help shape the ecosystem of collaborative models.

Appendix

Appendix A

Expanding the Reach of Federated Learning by Reducing Client Resource Requirements

A.1 Kashin's Representation

For reasons of space, we have relegated a more detailed discussion of Kashin's representation (see Section 2.2) to the Appendix. In this section, we briefly discuss Kashin's representation both from a theoretical (Section A.1.1) and practical (Section A.1.2) standpoints. Finally, we present some preliminary results that argue the potential of Kashin's representation to dominate over the random Hadamard transform with respect to the size vs. accuracy trade-off (Section A.1.3).

A.1.1 Theoretical Overview

The idea of using the classical results of Kashin (1977) to increase the robustness of coefficients to perturbations was first introduced by Lyubarskii & Vershynin (2010). Their result states that, given a tight frame satisfying a form of uncertainty principle, a weaker notion of the RIP (Candes et al., 2006), it is possible to convert the frame representation of every vector into the more robust *Kashin's representation*, whose coefficients will have the smallest possible dynamic range.

Error rates. Since the results of Suresh et al. (2017b) (who quantified the reduction in quantization error due to the Hadamard transform) rely on exactly this notion of dynamic range, and assuming the subsampled randomized Hadamard transform satisfies the uncertainty principle, Theorem 3.5 of Lyubarskii & Vershynin (2010) can be directly used as a drop-in replacement for Lemma 7 in Suresh et al. (2017b), removing the logarithmic dependence on dimension from Theorem 3 therein, matching the lower bounds. We do not provide the complete proof as, beyond drawing this connection, it does not imply any novelty whatsoever. However, an open question remains, as we are not aware of a result showing what are the parameters of the uncertainty principle guaranteed by the subsampled randomized Hadamard transform. They exist however, as the transform is known to satisfy the RIP (Foucart & Rauhut, 2013), which is a stronger notion.

A.1.2 Practical Considerations

In practice, given a tight frame, the algorithm for computing Kashin's representation is straightforward. It runs for n iterations, and takes parameters η, δ as input. In a single iteration, one first computes the frame coefficients, projects them onto a L_∞ ball, and reconstructs the error in the original domain. Another iteration proceeds starting with the reconstructed error and a smaller ball. We refer the reader to Lyubarskii & Vershynin (2010) for more details regarding η, δ and their relationship with the uncertainty principle.

In our work, we use the randomized Hadamard transform as the initial tight frame (see Section A.1.1 for details on why this is possible). We also run the algorithm for just $n = 2$ iterations (as very often this provides most of the benefit), fixed $\delta = 1$, and used a variant of the algorithm which yields an exact representation (omitting the L_∞ projection in the last iteration). Given this, the choice of η is irrelevant. The dominant part of the computation is then three applications of the fast Walsh-Hadamard transform, as opposed to a single one in Konečný et al. (2016)).

As a particular example, say we are to compress an 80-dimensional vector. We first pad the vector with zeros, so that its dimension is 128 (the closest larger power of 2). Then, we multiply the vector by a diagonal matrix with independent Rademacher random variables ($D \in \mathbb{R}^{128\times128}$), followed by the application of the fast Walsh-Hadamard transform ($H \in \mathbb{R}^{128\times128}$). The first 80 columns of the matrix HD correspond to the tight frame used to find the Kashin's representation. Nonetheless, we avoid representing this explicitly.

Finally, note that, if the initial dimension was a power of 2, we need to pad zeros to the next power of 2 in order to realize any benefit over just using the Hadamard transform.

A.1.3 Dominance over Hadamard

Given the theoretical properties of Kashin's representation, we hypothesize it should dominate the random Hadamard transform when it comes to the size vs. accuracy trade-off. A preliminary experiment to corroborate this hypothesis is the following:

1. We train an MNIST model until we get an accuracy of around 99.3%.
2. We compress the original model using some linear transform, some subsampling ratio and some number of quantization bits.
3. We decompress the model and evaluate both its new accuracy and its L_2 distance to the original model.
4. We repeat the previous two steps for different linear transforms (identity, random Hadamard transform and Kashin's representation), subsampling ratios (0.25, 0.5 and 1.0) and quantization bits (1, 2, 4, 8, 16).

An important detail is that, whenever we use Kashin's representation, we do a grid search over the best values for n (from 1 to 10) and η. However, δ is kept fixed as 1.

The results of this experiment are shown in Figure A.1. In the legend, R corresponds to rotation — I for identity, HD for randomized Hadamard, Kashin for Kashin based on the randomized Hadamard; and SR corresponds to subsampling ratio — the fraction of elements to be

kept non-zero. In the top row, the figure shows the relationship of the accuracy of the compressed model vs. the number of bits used for quantization, and vs. the model's size (in MB). In the bottom row, the L_2 error incurred is plotted against the same. It is very clear then that Kashin's representation does dominate the other two representations when it comes to the size vs. accuracy trade-off, making up the Pareto frontier for all combinations of subsampling ratio and quantization bits. Nevertheless, we did optimize over the parameters associated with Kashin's algorithm, something that does not need to be done for the random Hadamard transform. In Section A.1.2, we propose a set of values that worked well enough for our experiments, but further exploration on how to easily determine these values is in order.

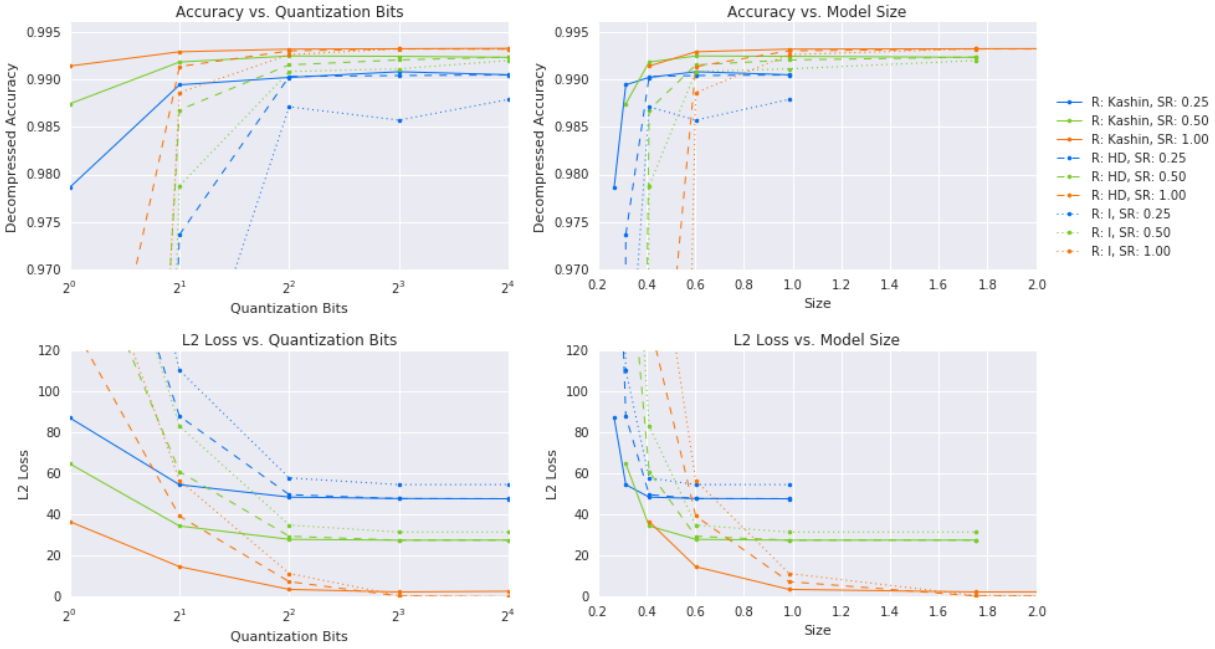

Figure A.1: Compressing an already trained MNIST model with linear transform + subsampling + uniform quantization.

A.2 MNIST Experimental Results

For reasons of space, we have relegated the experimental results using MNIST (see Section 2.3) (Section A.1) to the Appendix.

Figure A.2 shows the results of using our lossy compression on MNIST under the experimental setup presented in Section 2.3.2. Meanwhile, Figure A.3 shows the results of using *Federated Dropout* (see Section 2.3.3 for details). Finally, Figure A.4 shows the results of performing both lossy compression for downloads and uploads, as well as *Federated Dropout*, as described in Section 2.3.4.

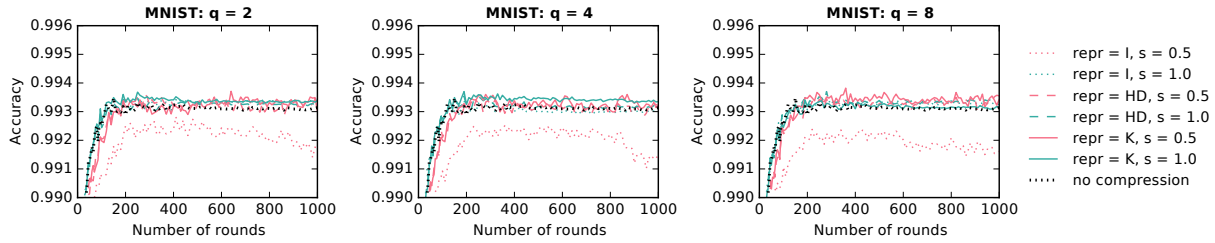

Figure A.2: Effect of varying our lossy compression parameters on the convergence MNIST.

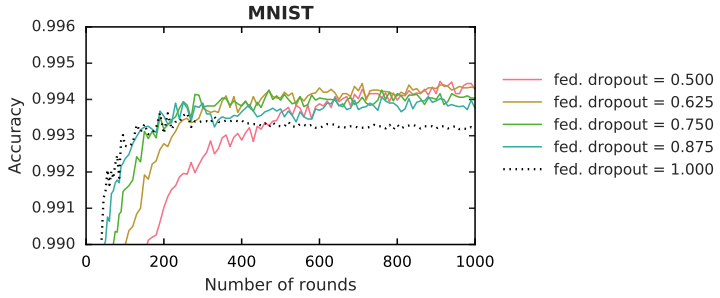

Figure A.3: Effect of varying the percentage of neurons *kept* in each layer on MNIST.

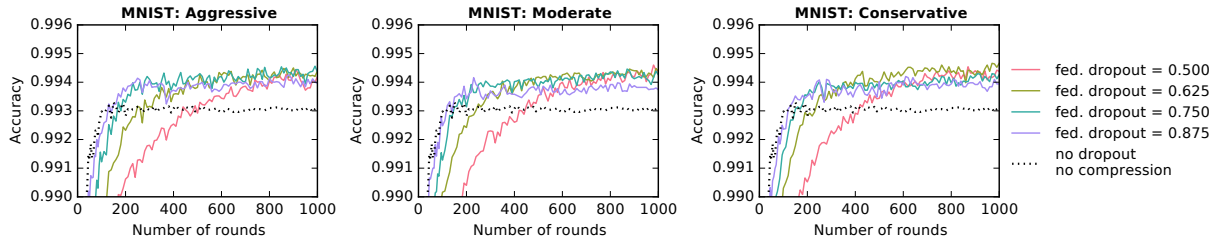

Figure A.4: Effect of using both lossy compression and *Federated Dropout* on MNIST.

Appendix B

Understanding Clinical Collaborations Through Federated Classifier Selection

We present additional details relating to the experiments presented in Chapter 3.2.

B.1 Tuning the Number of Neighbors in FRCLS

We expand on how to tune the number of neighbors k used to estimate $L_c(x, k)$. Through k we control how useful our estimates are: too high and we loose the local information on which FRCLS relies, too low and we overfit to noise. To avoid these scenarios, we propose a heuristic to tune k on our validation set.

For this tuning, we use $\ell_c(x, y) = \ell_c(c(x), y)$, where ℓ is the cross-entropy loss, as a surrogate for the utility of classifier c. This way, it is easy to differentiate two groups of instances: those for which $\ell_{c_E} < \ell_{c_L}$, and those for which $\ell_{c_E} \geq \ell_{c_L}$. If k is chosen correctly, then we expect the distribution of ρ_E in each one of these groups to not change drastically between validation and test sets. Our heuristic aims to minimize this change but instead uses two disjoint splits of the validation set.

To measure changes in distribution, we use the Rényi divergence (Póczos & Schneider, 2011). For each group of instances defined above, we explore a grid of possible values for k and plot the resulting divergences. Finally, we find the *knee* of each curve and choose the highest k between them. In our experiments we explore the range of values $k = \{2, 7, 15, 50, 100, 150, 500, 1000\}$ and end up using $k = 100$.

B.2 Data Description for Early Prediction of Sepsis

We present additional details on the data used in the experiments presented in Chapter 3. Table B.1 shows the dataset size for each hospital system. Tables B.2 and B.3 describe the features in our data. Finally, Table B.4 provides a complete description of the instances we use to exemplify how to use FRCLS's rules.

Hospital System	Number of Instances
A	$31,253$
B	$24,579$

Table B.1: Number of instances in each hospital system.

Variable	System A	System B
AST	43.0 (24.0/91.0)	33.0 (21.0/67.0)
Age	65.1 (52.4/75.8)	62.0 (50.0/72.0)
Alkalinephos	78.0 (56.0/116.0)	70.0 (53.0/99.0)
BUN	19.0 (13.0/32.0)	19.0 (12.0/32.0)
Base Excess	0.0 (-2.0/3.0)	-3.5 (-5.7/-0.8)
Bilirubin Total	0.7 (0.4/1.5)	0.9 (0.6/1.5)
Calcium	8.3 (7.8/8.7)	8.3 (7.5/8.8)
Chloride	106.0 (102.0/109.0)	106.0 (103.0/110.0)
Creatinine	0.9 (0.7/1.4)	1.0 (0.8/1.7)
DBP	59.0 (52.0/68.0)	64.0 (55.5/74.0)
FiO2	0.5 (0.4/0.5)	0.4 (0.4/0.6)
Glucose	125.0 (105.0/152.0)	123.0 (104.0/150.0)
HCO3	24.0 (22.0/27.0)	22.1 (20.1/24.7)
HR	87.0 (75.0/100.0)	86.0 (74.0/99.0)
Hct	30.5 (27.7/34.0)	30.8 (26.5/35.8)
Hgb	10.4 (9.3/11.6)	10.0 (8.6/11.7)
Lactate	1.4 (1.1/2.1)	1.6 (1.2/2.3)
MAP	77.0 (68.0/87.0)	83.0 (73.0/96.0)
Magnesium	2.0 (1.8/2.2)	2.0 (1.9/2.3)
O2Sat	98.0 (96.0/99.0)	98.0 (95.0/99.5)
PTT	31.0 (27.0/38.2)	31.6 (28.2/38.1)
PaCO2	40.0 (36.0/45.0)	38.0 (34.0/44.0)
Phosphate	3.3 (2.7/4.1)	3.4 (2.7/4.2)
Platelets	193.0 (137.0/267.0)	180.0 (123.0/248.0)
Potassium	4.0 (3.7/4.4)	4.0 (3.7/4.4)
Resp	19.0 (16.0/23.0)	18.0 (16.0/22.0)
RR	0.7 (0.6/0.8)	0.7 (0.6/0.8)
SBP	118.0 (104.0/135.0)	122.0 (106.0/141.5)
SaO2	97.0 (93.0/98.0)	97.5 (95.7/98.8)
Temp	37.1 (36.6/37.6)	36.8 (36.4/37.5)
WBC	11.3 (8.3/15.0)	10.3 (7.5/14.1)
pH	7.4 (7.4/7.4)	7.4 (7.3/7.5)

Table B.2: Numerical features in our data. We show the median (Q1/Q3) for each one of our hospital systems.

Variable (Value)	System A	System B
Gender (0)	40.48%	44.47%
Gender (1)	59.52%	55.53%
Unit1 (0)	18.85%	36.88%
Unit1 (1)	29.29%	34.42%

Table B.3: Categorical features in our data. We show the percentage of the specified value in each one of our hospital systems. Unit1 is an administrative reference to a medical ICU (as opposed to a surgical one). A Gender of 0 refers to female.

Variable	Patient A	Patient B
Age	69	54
Gender	1	1
AST	-	23
Alkalinephos	-	58
BUN	13	13
Base Excess	-	-3.2
Bilirubin total	-	2.1
Calcium	8.6	4.39
Chloride	110	113
Creatinine	1	0.8
DBP	43	60
FiO2	-	1
Glucose	140	116
HCO3	27	22.4
HR	51	80
Hct	38.5	31.7
Hgb	13.1	10.9
Lactate	-	1.25
MAP	63	75
Magnesium	2.4	1.9
O2Sat	97	98
PTT	28.2	38.4
PaCO2	-	42
Phosphate	2.5	2.1
Platelets	107	140
Potassium	4	4.4
Resp	16	18
RR	1.17647	0.75
SBP	123	126
SaO2	-	95.9
Temp	36.06	37.4
Unit1	-	1
WBC	25.2	14.9
pH	-	7.34
True Label	1	0

Table B.4: Complete list of features for instances whose predictions get flipped with the use of c_E. We present one instance per hospital system.

Appendix C

Using Machine Learning to Support Transfer of Best Practices in Healthcare

In Table C.1, we show description for the DRG that were selected for the experiments in Chapter 4.

DRG	Description	# of Claims	
		Org. A	Org. B
189	Pulmonary Edema & Respiratory Failure	2,989	4,665
194	Simple Pneumonia & Pleurisy W Cc	1,883	2,206
291	Heart Failure & Shock W Mcc	4,624	5,593
292	Heart Failure & Shock W Cc	2,775	4,705
392	Esophagitis, Gastroent & Misc Digest Disorders W/o Mcc	3,837	4,432
470	Major Joint Replacement Or Reattachment Of Lower Extremity W/o Mcc	8,299	19,728
603	Cellulitis W/o Mcc	3,369	2,832
690	Kidney & Urinary Tract Infections W/o Mcc	2,109	1,729
765	Cesarean Section W Cc/mcc	4,387	13,019
766	Cesarean Section W/o Cc/mcc	7,840	10,278
774	Vaginal Delivery W Complicating Diagnoses	3,165	10,903
775	Vaginal Delivery W/o Complicating Diagnoses	22,531	44,080
871	Septicemia Or Severe Sepsis W/o Mv 96+ Hours W Mcc	12,375	23,278
872	Septicemia Or Severe Sepsis W/o Mv 96+ Hours W/o Mcc	4,927	11,710

Table C.1: Details for each selected DRG.

Appendix D

Encoding Expert Knowledge Into On-Device Data Using Weak Supervision

D.1 Datasets and Models

We provide a description of the datasets and models used in our work. We use federated versions of three different datasets: the Amazon product reviews dataset (Ni et al., 2019), the IMDb movie reviews dataset (Maas et al., 2011) and the Massachusetts Institute of Technology – Beth Israel Hospital Arrhythmia Database (MIT BIH) dataset (Moody & Mark, 2001; Goldberger et al., 2000). Statistics on the number of clients and examples in the different splits of these datasets are given in Table D.1.

	Num. Examples	Num. Clients	Mean Examples per Client (std)	Fraction of Positive Class
			Amazon	
Train	119,725	738	162.22 (73.36)	0.54
Val	20,090	123	-	0.54
Test	60,366	369	-	0.55
			IMDb	
Train	20,000	1000	20.0 (0.0)	0.50
Val	5,000	-	-	0.49
Test	25,000	-	-	0.50
			MIT BIH	
Train	21,008	36	583.55 (461.72)	0.58
Val	2,939	4	-	0.72
Test	4,153	8	-	0.60

Table D.1: Details for datasets and partitions used in our experiments. We treat the validation and test partition as if it were centralized in the server.

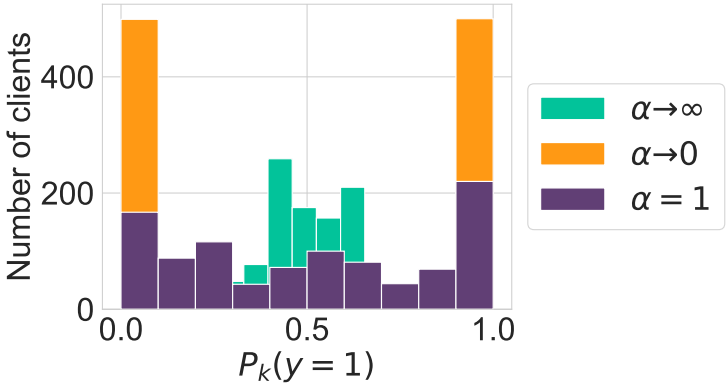

Figure D.1: Histogram of local class balances $P_k(y)$ as we vary α for the IMDb dataset.

D.1.1 Amazon

We use a subset of the Amazon product reviews dataset (Ni et al., 2019), solving a binary sentiment classification task. To construct our federated dataset, we first aggregate all categories with more than $100k$, and constructed each client k based on the available reviewer ids. We then sampled reviewers in ascending order based on quantity $|P_k(y = 1) - 0.5|$ until we had more than $200k$ reviews. Intuitively, we looked for reviewers with class balances close to 0.5. Finally, we performed a $60/10/30$ train/val/test split. We featurize this data using a pre-trained open-source sentence transformer (Reimers & Gurevych, 2019; Sentence Transformers, 2019), which outputs a feature vector of 768 dimensions. Our end model is a multilayer perceptron with two hidden layers of size 20 and RELU activations.

D.1.2 IMDb

We use the IMDb movie reviews dataset (Maas et al., 2011). This dataset has $25k$ training examples and $25k$ test examples. We further split the training set into $20k$ examples for training and $5k$ examples for validation, and create $1k$ training clients by splitting the reviews in the training set uniformly at random. We use the same featurization and end model as for the Amazon dataset. Finally, in Figure D.1, we show the distribution of local class balances $P_k(y)$ as we vary the parameter α (see Section 5.5.2).

D.1.3 MIT BIH

The MIT BIH dataset comprises of 48 half-hour excerpts of two-lead ambulatory ECG recordings from 47 subjects. The dataset contains beat-level annotations for a wide range of heart beats, ranging from normal to arrhythmia (e.g., left bundle branch block, premature ventricular contraction, etc). Since detecting all varieties of arrhythmia is challenging and not the primary goal of our study, we solved a simpler binary classification task of discriminating normal heart beats from arrhythmias.

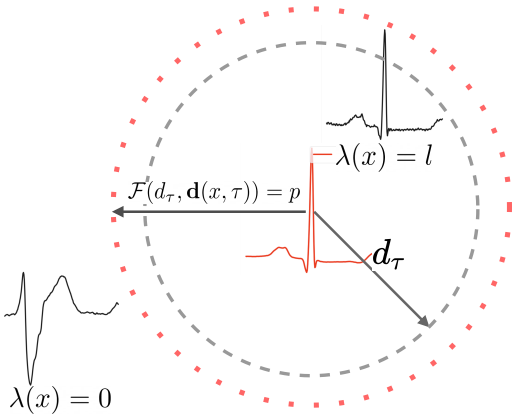

Figure D.2: A time-series LF λ is parameterized using a three-tuple (τ, d_τ, l). The template τ shown in red is the centroid of a cluster, represented using the dashed lines $--$. d_τ is the radius of the cluster and corresponds to the distance of the cluster member farthest from τ. Depending on the probability threshold p, this LF λ will label data points x as belonging to class l, or it will abstain from voting. This labeling threshold is denoted by the outer concentric circle $\cdots$.

On the MIT BIH dataset, we treat each patient as a different client. The patients used in each of the partitions in Table D.1 are as follows:

Validation:	102, 115, 123, 202
Test:	101, 105, 114, 118, 124, 201, 210, 217
Train:	All other patients

In Figure D.2, we illustrate the parameterization of the LFs we use for this dataset. As an end model, we train a one-dimensional convolutional neural network. Figure D.10 shows the definition of the model we use. As input into our model, we use the Modified Limb lead II (MLII) obtained by placing electrodes on the chest, as is done in prior work (Goswami et al., 2021). We output a prediction for each window of 256 samples (sampled at 360Hz) around peaks given by previous preprocessing. Finally, we use early stopping based on our validation ROC AUC to avoid overfitting when training WeaSEL (Figure 5.5, Figure 5.7) and as the expert inspects the LFs (Figure 5.7).

D.1.4 Additional Models

Our label model is always a multi-layer perceptron with two hidden layers of size 20 and ReLU activations. When training this model, we set the class balance $P(y)$ to 0.5. Models $h_k : \tau_k(\lambda) \rightarrow u$ are multilayer perceptrons with two hidden layers of size 10 and ReLU activations.

D.2 Experiment Hyperparameters

We describe the hyperparameters used for our experiments in Section 5.5. Section D.2.1 presents the parameters used in the experiments presented in Table 5.1, Figure 5.4, Figure 5.7 and Figure 5.8. Meanwhile, Section D.2.2 presents the parameters used in Figure 5.5 and Figure 5.7.

D.2.1 Automatic Mining of LFs

When generating time-series LFs as described in Section 5.3, we use the following parameters:

Probability threshold (p) :	0.2
Number of clusters in client k :	3 `if` $n_k < 20$ `else` $\lfloor n_k/5 \rfloor$
Number of clusters in server :	500

When running Algorithm 1 to mine LFs, we train model h_k in each client k with the Adam optimizer (Kingma & Ba, 2014), using early stopping on the training loss. The hyperparameters that we use for this algorithm are as follow:

Delta (δ) :	0.05
Clients per round (K) :	10
Batch size :	64
Maximum number of epochs :	200
Learning rate :	1e−3
Weight decay :	1e−4

Due to the difference in the total number of clients between datasets, we use a different number of rounds R per data modality. For text, we set R to 10, while for time-series we use 1.

D.2.2 Training of the PWS Model

Throughout these experiments, we compare the performance of training a federated version of `WeaSEL` using `FedAvg`, `FedProx` and `FedAdam`. For `FedAvg` and `FedAdam`, the client optimizer is mini-batch SGD, while for `FedProx` it includes a proximal term with weighted by $\mu > 0$. For all algorithms, we tune the hyperparameters using random search, exploring 20 sets of parameters and choosing the set with the best ROC AUC on the validation dataset. We perform this tuning once using the pre-curated set of LFs presented in Appendix D.3.

The hyperparameters that we explore are the following:

$\log_{10}$(client learning rate) :	Unif$[-2, -1]$
Temperature of `WeaSEL` model :	Unif$[10, 25]$
Number of client epochs :	Unif$\{1, 3, 5\}$

For `FedProx` we tune μ in Unif$\{$1e−3, 1e−2, 1e−1, 1$\}$. For `FedAvg` and `FedProx`, we explore a $\log_{10}$(server learning rate) in Unif$\{-2, -1, 0\}$. For `FedAdam`, we explore the same

hyperparameter in the range Unif$[-5, -4]$ for Amazon and IMDb, and in the range Unif$\{-4, -3, -2\}$ for MIT BIH. We set the client and server momentums to 0.9, the batch size to 64 and the clients per round to 10.

In Table D.2 we present the hyperparameters chosen after tuning.

	client lr	server lr	temperature	epochs	μ
			FedAvg		
Amazon	3.35e−2	0.01	14.37	5	-
IMDb	4.15e−2	1.00	24.75	1	-
MIT BIH	2.66e−2	0.10	18.20	3	-
			FedProx		
Amazon	3.70e−2	0.01	24.54	3	0.01
IMDb	4.60e−2	0.10	16.38	3	0.01
MIT BIH	6.81e−2	1.00	23.64	3	1.00
			FedAdam		
Amazon	8.12e−2	1.81e−5	19.94	3	-
IMDb	4.82e−2	4.07e−5	22.50	5	-
MIT BIH	2.66e−2	1.00e−3	18.20	3	-

Table D.2: Hyperparameters chosen after performing random search over the grids presented in Appendix D.2.2.

D.2.3 Baselines

We tune the hyperparameters for two baselines in our experiments: a fully supervised baseline that we train using FedAvg, and a centralized version of WeaSEL. For both baselines, we tune their hyperparameters using random search, exploring 10 sets of parameters and choosing the set with the best ROC AUC on the validation dataset. For the WeaSEL baseline, we perform this tuning using the pre-curated set of LFs presented in Appendix D.3.

For our supervised (FedAvg) baseline, the hyperparameters that we explore are the following:

$$\begin{aligned}
\log_{10}(\text{client learning rate}): &\quad \text{Unif}[-4, -2] \\
\log_{10}(\text{server learning rate}): &\quad \text{Unif}\{-2, -1, 0\} \\
\text{Number of client epochs}: &\quad \text{Unif}\{1, 3, 5\} \\
\text{Client momentum}: &\quad 0.9 \\
\text{Server momentum}: &\quad 0.9 \\
\text{Batch size}: &\quad 64 \\
\text{Clients per round}: &\quad 10
\end{aligned}$$

For our centralized (WeaSEL) baseline, the hyperparameters that we explore are the following:

$$
\begin{aligned}
\log_{10}(\text{learning rate}): &\quad \text{Unif}[-4, -3] \\
\text{Temperature of WeaSEL model}: &\quad \text{Unif}[10, 25] \\
\text{Momentum}: &\quad 0.9 \\
\text{Number of epochs}: &\quad 200 \\
\text{Batch size}: &\quad 64
\end{aligned}
$$

In Table D.3 we present the hyperparameters chosen after tuning.

Supervised (FedAvg)

	client lr	**server lr**	**epochs**
Amazon	5.26e−3	1.00	3
IMDb	5.26e−3	1.00	3
MIT BIH	2.15e−3	1.00	1

Centralized (WeaSEL)

	lr	**temperature**
Amazon	3.06e−4	18.89
IMDb	8.75e−4	22.99
MIT BIH	8.75e−4	22.99

Table D.3: Hyperparameters chosen after performing random search over the grids presented in Appendix D.2.3.

D.3 Labeling Functions used for Federated Weasel

For our experiments in Section 5.5.2, we use a set of pre-curated LFs. For Amazon and IMDb, we use the LFs reported by Boecking et al. (2020), which correspond to examples of heuristics that real users found useful when asked for the same type of feedback as the one described in this work. Meanwhile, for MIT BIH, we adopted an automated procedure to identify these LFs: using the method proposed by Boecking et al. (2020) in a centralized setting, using the same oracle as our experiments. The LFs used are detailed below.

- **Amazon:**
  - Positive: amazing, awesome, beautiful, beautifully, best, captivating, comfy, compliments, delightful, durable, easy, excellent, expected, fantastic, favorite, gorgeous, great, interesting, love, loves, perfect, perfectly, pleasantly, stars, strong, value, wonderful.
  - Negative: awful, bad, beware, boring, crap, disappointing, garbage, horrible, joke, junk, mess, money, poor, poorly, refund, sent, terrible, unusable, useless, waste, wasted, worse, worthless, worst, yuck, zero.
- **IMDb:**
  - Positive: amazing, art, beautiful, beautifully, breathtaking, brilliant, captures, delight, delightful, enjoyed, excellent, masterpiece, fantastic, favorites, finest, flawless, intelligent, joy, light, perfect, perfection, refreshing, superb, superbly, terrific, underrated, wonderful, wonderfully.
  - Negative: atrocious, awful, bad, boring, crap, decent, dreck, dull, failed, horrible, lame, laughable, lousy, mistake, pointless, poor, reason, redeeming, ridiculous, stinker, stupid, terrible, unfunny, unwatchable, waste, worst.
- **MIT BIH:** Figure D.3 shows the LFs we use for our MIT-BIH experiments.

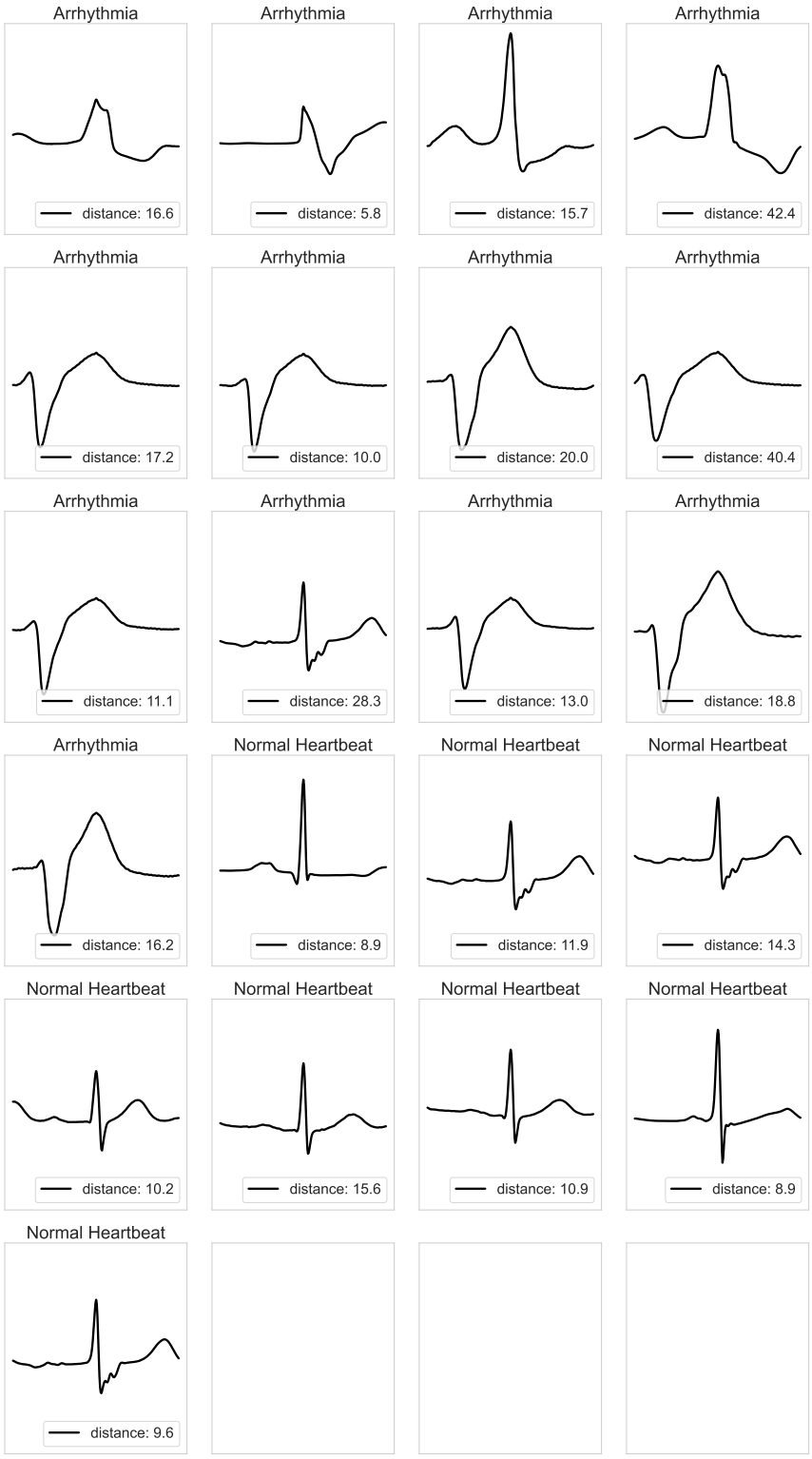

Figure D.3: Visualization of the LFs used when training WeaSEL in a federated manner, using the MIT BIH dataset.

D.4 Labeling Function Seeds

In our experiments, we initialize Algorithm 1 at time-step 0 with a set S of seed LFs. We use four seeds, two for each class, with a training accuracy above 0.7 and thus labeled with $u = 1$. Furthermore, for our text datasets, in each repetition, we randomly sample four additional seeds in hopes of discovering LFs marked with $u = 0$. We don't perform this random sampling for the time-series modality as it heavily increased the variance of the ROC AUC at time-step 0.

The seeds that were used throughout the experiments were:

- **Amazon:**
  - Positive: adorable, thoughtful.
  - Negative: stereotypical, horrible.
- **IMDb:**
  - Positive: adorable, witty.
  - Negative: stereotypical, hated.
- **MIT BIH:** Figure D.4 shows LFs we use as seeds for our MIT-BIH experiments. These LFs were chosen by the authors from the pool in Appendix D.3 based on visual inspection.

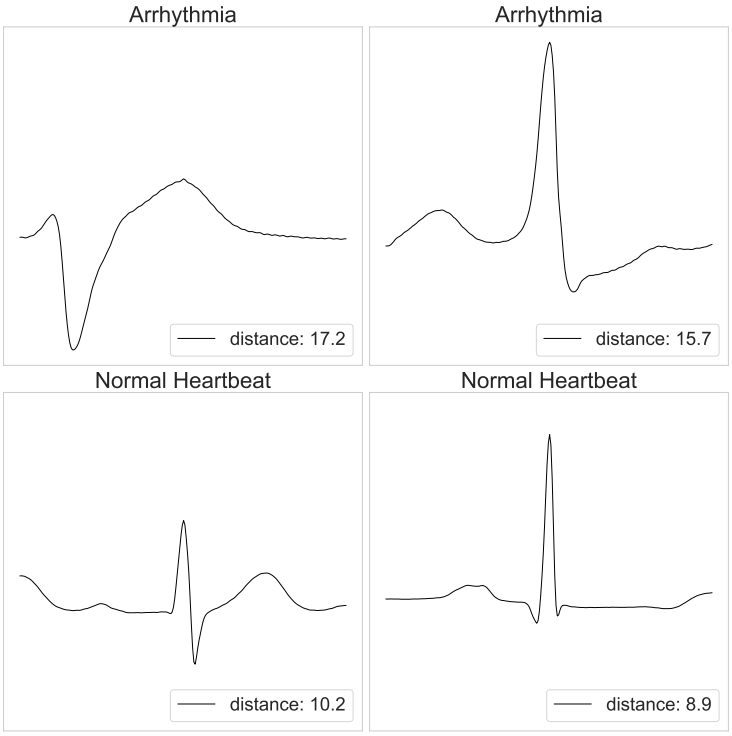

Figure D.4: Visualization of the seeds used in our MIT BIH experiments. In our set up, arrhythmia corresponds to the positive class.

D.5 Examples of Inspected Labeling Functions

We show examples of LFs considered useful by our simulated oracle, i.e., labeled with $u = 1$, on our three datasets. For Amazon and IMDb, we show the five heuristics most frequently annotated as useful, and break ties at random. For MIT BIH, we show the most accurate LFs found useful by the oracle, and break ties by selecting candidates that are visually dissimilar. The list of example heuristics is presented below:

- **Amazon:**
  - WSHFL:
    - Positive: size, love, recommend, perfect, highly.
    - Negative: poor, worse, boring, total, worst.
  - Greedy:
    - Positive: pastiche, pointofview, preschool, leveling, elm.
    - Negative: inarticulate, purchases, catnip, dart, selfabsorbed.
  - Random:
    - Positive: guitars, cliffhanger, break, domicile, define.
    - Negative: capable, tend, whichfacilitate, itunes, notable.
- **IMDb:**
  - WSHFL:
    - Positive: shows, performance, perfect, fun, excellent.
    - Negative: money, annoying, badly, awful, lame.
  - Greedy:
    - Positive: wordplay, caps, adoree, accelerated, ardour.
    - Negative: cadet, appendage, accuses, blueprints, beanies.
  - Random:
    - Positive: poltergeist, conversation, brawny, damages, 30mins.
    - Negative: critiquing, approved, effortless, completely, banner.
- **MIT BIH:**
  - WSHFL: We plot three labeling functions inspected by the expert using WSHFL in Figure D.5.
  - Greedy: We plot three labeling functions inspected by the expert using our greedy baseline in Figure D.6.
  - Random: We plot three labeling functions inspected by the expert using our random baseline in Figure D.7.

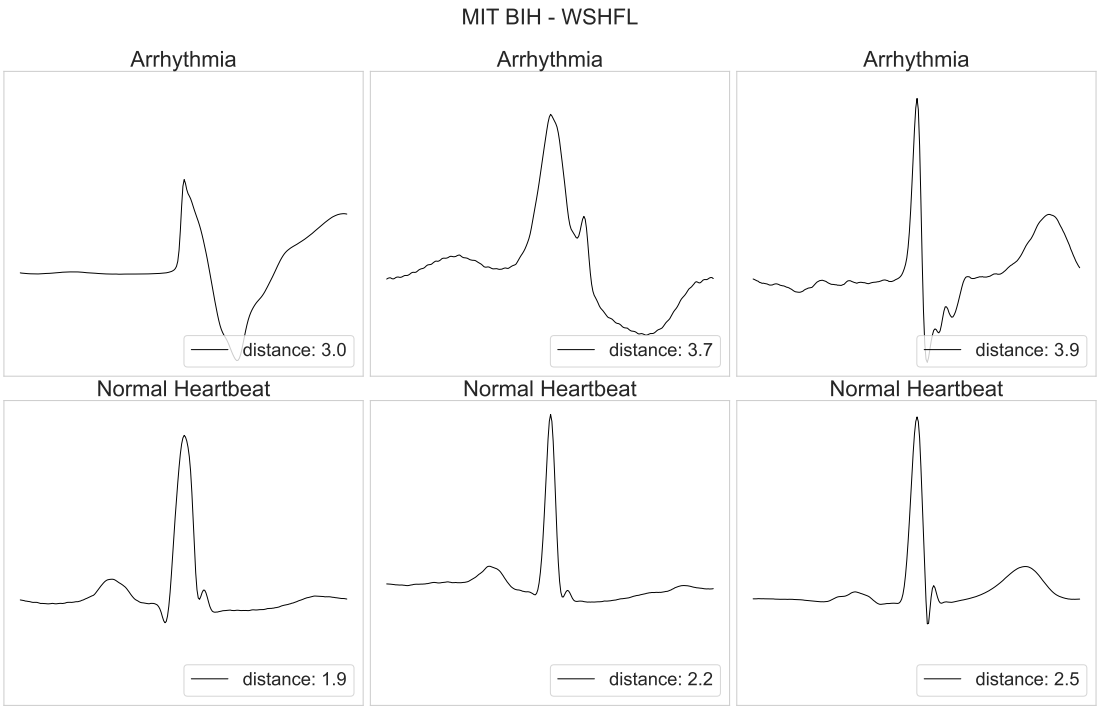

Figure D.5: Visualization of the most accurate candidates inspected by the expert when using WSHFL with the MIT BIH dataset. We present the three top candidates per class and break ties by selecting candidates that are visually dissimilar.

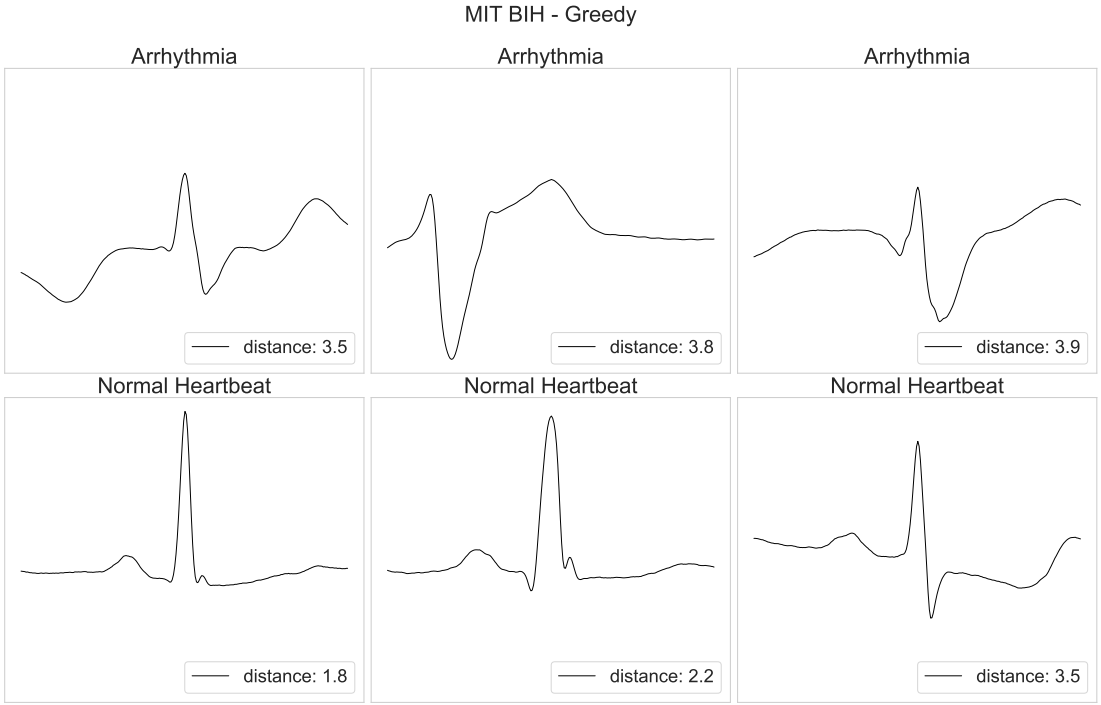

Figure D.6: Visualization of the most accurate candidates inspected by the expert when using our greedy baseline with the MIT BIH dataset. We present the three top candidates per class and break ties by selecting candidates that are visually dissimilar.

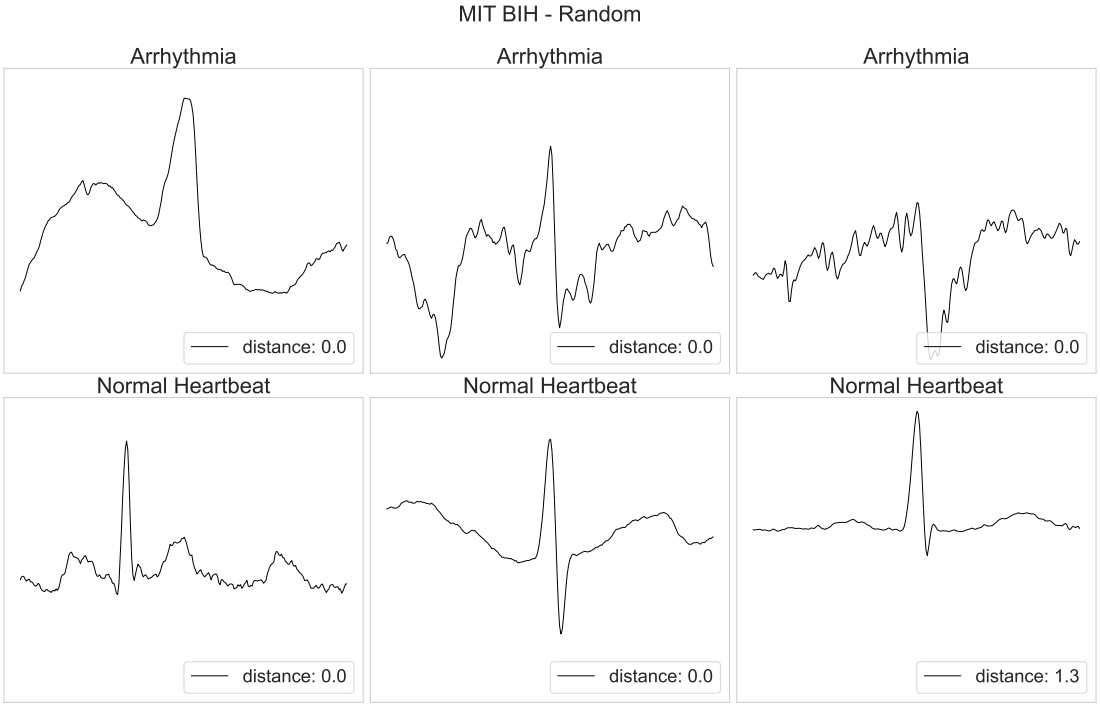

Figure D.7: Visualization of the most accurate candidates inspected by the expert when using our random baseline with the MIT BIH dataset. We present the three top candidates per class and break ties by selecting candidates that are visually dissimilar.

D.6 Ablations

We investigate the behavior of our end-to-end WSHFL experiments with experts of different confidence levels. To do this, we vary the threshold at which our oracle labels a LF with $u = 1$. In Figure D.8, we observe how WSHFL is robust to experts with different confidence levels, starting to degrade for Amazon and IMDb once the expert starts accepting LFs with accuracies close to random.

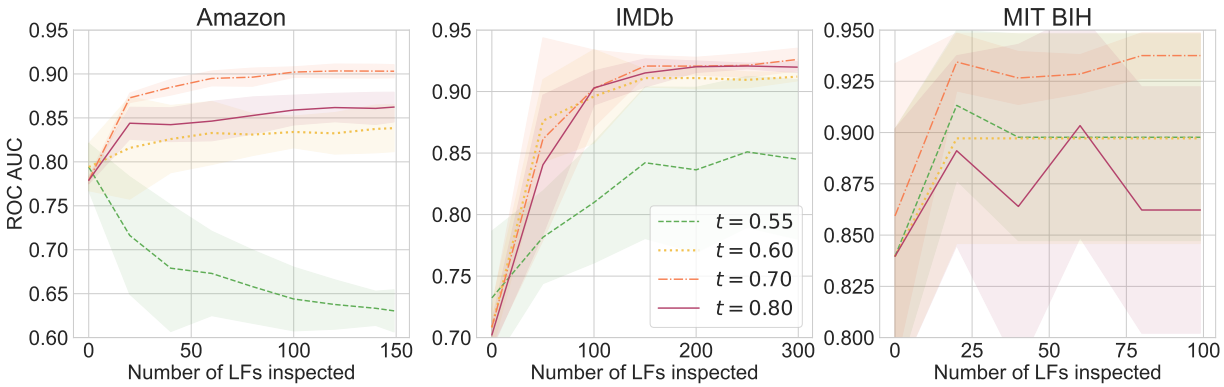

Figure D.8: Results of our ablation experiments on WSHFL, where we vary the threshold of the oracle we use as an expert. We observe how our experiments are robust to a range of thresholds, yet may start to degrade when experts accept LFs too close to random.

D.7 Proposed Candidates Distribution

In Figure D.9, we plot the accuracies and coverages for all the aggregated candidates at the server. This illustrates the distribution of LFs over which our method and baselines are sampling over. For all datasets, we intuitively observe a bimodal distribution of LFs based on their accuracy. This is because we exhaustively assign all classes to keyword/templates to generate LFs, hence for every accurate LF, we also have an equally inaccurate LF candidate. However, we found that the distribution of the accuracy of candidate LFs is different for text and time-series datasets. In particular, time-series LF candidates either had high or low accuracy, with few intermediate values.

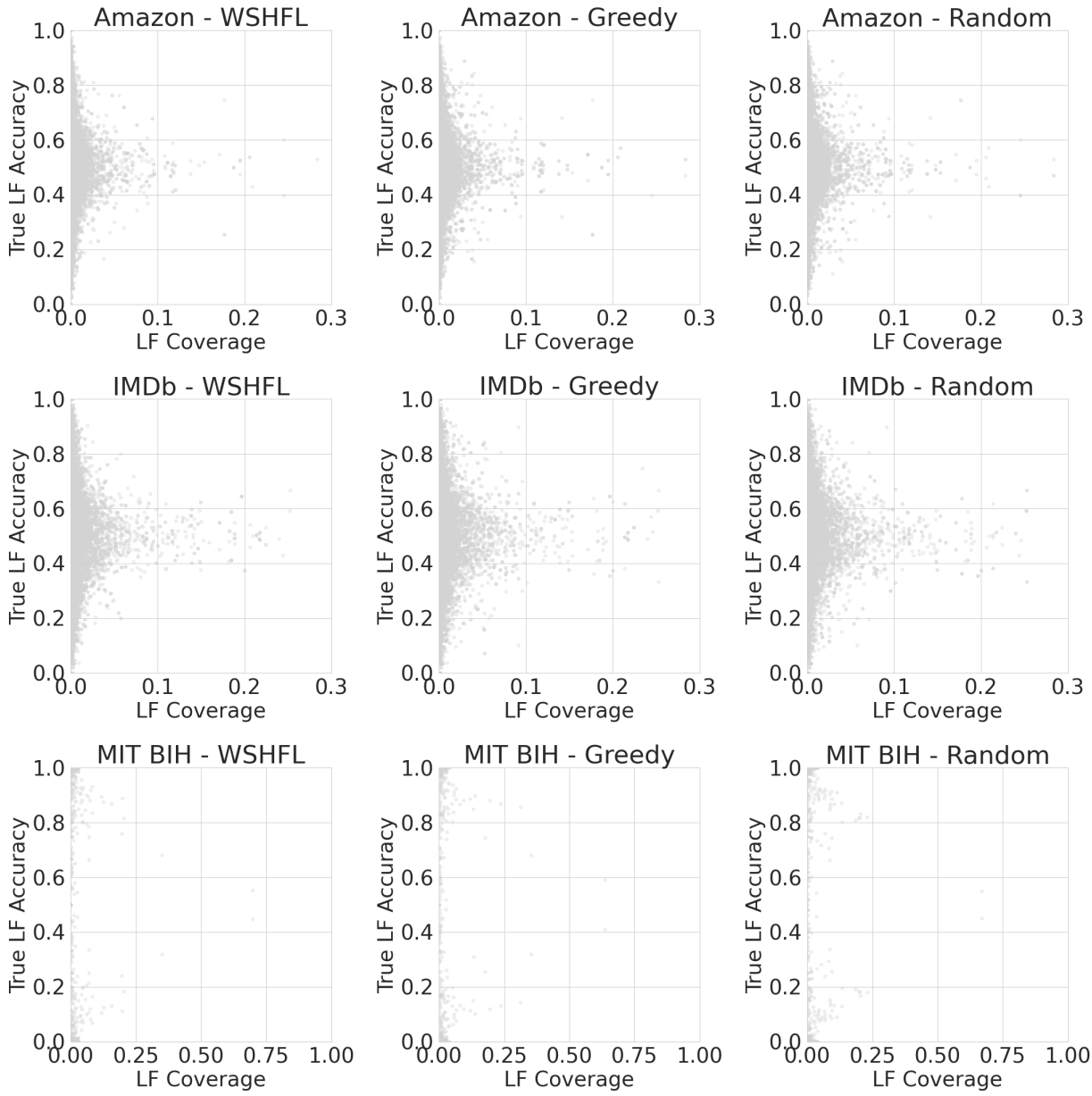

Figure D.9: Training accuracies vs. coverages for the LFs aggregated at the server. We plot the candidates for the last time-step in one repetition chosen at random.

```
class ConvNet(nn.Module):
  def __init__(self, input_dim, output_dim):
    super().__init__()
    self.output_dim = output_dim
    self.input_dim = input_dim

    self.conv = nn.Sequential(*self.build_conv())
    self.output = nn.Sequential(*self.build_output())

  def build_conv(self):
    return [
      nn.Conv1d(
        in_channels=self.input_dim, out_channels=4,
        kernel_size=8, stride=1, padding='same'),
      nn.BatchNorm1d(num_features=4),
      nn.ReLU(inplace=True),
      nn.Conv1d(
        in_channels=4, out_channels=4,
        kernel_size=5, stride=1, padding='same'),
      nn.BatchNorm1d(num_features=4),
      nn.ReLU(inplace=True),
      nn.Conv1d(
        in_channels=4, out_channels=4,
        kernel_size=3, stride=1, padding='same'),
      nn.BatchNorm1d(num_features=4),
      nn.ReLU(inplace=True)]

  def build_output(self):
    return [
      nn.Linear(in_features=4, out_features=self.output_dim),
      nn.Softmax(dim=1)]

  def forward(self, x):
    output_conv = self.conv(torch.unsqueeze(x, dim=1))
    output_avg = nn.AvgPool1d(
      kernel_size=output_conv_block.shape[2],
      stride=1)(output_conv_block).squeeze()
      return self.output(output_avg)
```

Figure D.10: Definition of the one-dimensional convolutional network used for the MIT BIH dataset.

Appendix E

LEAF: A Benchmark for Cross-Device Settings

E.1 Synthetic Dataset

Our synthetic dataset (introduced in Section 6.1) is inspired by the one presented in Li et al. (2020c), but has possible additional heterogeneity designed to make current meta-learning methods (such as *Reptile* Nichol et al. (2018)) fail. The high-level goal is to create tasks whose true models are (1) task-dependant, and (2) clustered around more than just one center.

To start, the user must input the desired number of devices $T \geq 1$ and a vector $(p_1, \ldots, p_k)$ such that $p_j > 0, j \in 1, \ldots, k$ and $\sum_{j=1}^{k} p_j = 1$. As preparation to generate the tasks:

1. Sample cluster means $\mu_j \in \mathbb{R}^s, j \in 1, \ldots, k$. To do this, draw $\mu_j \sim N(B_j, I), B_j \sim N(0, I)$.
2. Draw matrix $Q \in \mathbb{R}^{d+1 \times s}$ by sampling $Q \sim N(0, I)$.
3. Create diagonal matrix Σ such that $\Sigma_{i,i} = i^{-1.2}$.

Now, for each task $t \in 1, \ldots, T$:

1. Sample a cluster center μ_t according to the input probabilities $(p_1, \ldots, p_k)$.
2. Draw $u_t \sim N(\mu_t, I)$ and set $w_t = Qu_t, w_t \in \mathbb{R}^{d+1}$.
3. Now, draw m_t from a log-normal distribution with mean 3 and sigma 2. We then set the number of samples $n_t = \min(m_t + 5, 1000)$ (to put a lower and an upper bound on the number of samples per task).
4. Sample $v_t \sim N(C_t, I), C_t \sim N(0, I)$.
5. Now, for $i \in 1, \ldots, n_t$, sample $x_t^i \in \mathbb{R}^d$ by drawing $x_t^i \sim N(v_t, \Sigma)$.
6. Finally, set $y_t^i = \arg\max(\text{sigmoid}(w_t x_t^i + N(0, 0.1 \cdot I)))$ after adding the necessary padding to x_t^i to account for the intercept.

E.2 Experiment Details

In this section, we provide details for the experiments presented in Section 6.2.

Shakespeare convergence. For the experiment presented in Figure 6.2, we subsample 118 devices (around 5% of the total) in our Shakespeare data. Our model first maps each character to an embedding of dimension 8 before passing it through an LSTM of two layers of 256 units each. The LSTM emits an output embedding, which is scored against all items of the vocabulary via dot product followed by a softmax. We use a sequence length of 80 for the LSTM. We evaluate using `AccuracyTop1`. We use a learning rate of 0.8 and 10 devices per round for all experiments.

Statistical and systems analyses. For all the Sent140 experiments presented in Figure 6.3, we use a bag of words model with logistic regression, and a learning rate of $3 \cdot 10^{-4}$. For the FEMNIST experiments in the same figure, we subsample 5% of the data, and use a model with two convolutional layers followed by pooling, and a final dense layer with 2048 units. We use a learning rate of $4 \cdot 10^{-3}$ for FedAvg and of $6 \cdot 10^{-2}$ for minibatch SGD.

Additional pipelines. For the experiments presented in Table 6.2 we use a split of 60% training, 20% validation and 20% test per user, and report results on the test set. The hyperparameters that vary per experiment are the following:

- For the CelebA experiments, we use 10% of the total clients and the same model we described above for FEMNIST. For the local models, each device explored learning rates in $[0.1, 0.01, 0.001, 0.0001]$. The FedAvg model uses 10 clients per round for 100 rounds, training locally for one epoch with a batch size of 5, and a best learning rate of 0.001. Both results are averaged over 5 runs.
- For the experiments with the Synthetic dataset, we use $1,000$ devices, only one cluster, 60 features and 5 classes. Our model is a perceptron with sigmoid activations. For the local models, each device explored learning rates in $[10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 10^2, 10^3]$. The FedAvg model used 10 clients per round for 100 rounds, trained locally for one epoch with a batch size of 5, and found a best learning rate of 0.1.
- For the Reddit experiments, we use 819 devices and a model similar to the one we described for Shakespeare. The main differences are: the size of the embedding is now 200, and we build the vocabulary from the tokens in the training set with a fixed length of $10,000$. We use a sequence length of 10, evaluate using `AccuracyTop1` and consider all predictions of the unknown and padding tokens as incorrect. For the global iid model, we train for 3 epochs over all the devices' data using a learning rate of $4 \cdot \sqrt{2}$. For FedAvg, we use 10 clients per round for 100 rounds, training locally for one epoch using a batch size of 5. We use a learning rate of 8. Both results are averaged over 5 runs.
- For the FEMNIST experiments we use the same model as described before and run each algorithm for $1,000$ rounds, use 5 clients per round, a local learning rate of 10^{-3}, a training mini-batch size of 10 for 5 mini-batches, and evaluate on an unseen set of test devices. Furthermore, for *Reptile* we use a linearly decaying meta-learning rate that goes from 2 to 0, and evaluate by fine-tuning each test device for 50 mini-batches of size 5.

Bibliography

Mohammed Adnan, Shivam Kalra, Jesse C Cresswell, Graham W Taylor, and Hamid R Tizhoosh. Federated learning and differential privacy for medical image analysis. *Scientific reports*, 12 (1):1–10, 2022.

Arvind Agarwal, Samuel Gerber, and Hal Daume. Learning multiple tasks using manifold regularization. In *Advances in neural information processing systems*, 2010.

Samiul Alam, Luyang Liu, Ming Yan, and Mi Zhang. Fedrolex: Model-heterogeneous federated learning with rolling sub-model extraction. *Advances in Neural Information Processing Systems*, 35:29677–29690, 2022.

Ehab A AlBadawy, Ashirbani Saha, and Maciej A Mazurowski. Deep learning for segmentation of brain tumors: Impact of cross-institutional training and testing. *Medical physics*, 45(3): 1150–1158, 2018.

Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. Qsgd: Communication-efficient sgd via gradient quantization and encoding. *Advances in neural information processing systems*, 30, 2017.

Dan Alistarh, Torsten Hoefler, Mikael Johansson, Nikola Konstantinov, Sarit Khirirat, and Cédric Renggli. The convergence of sparsified gradient methods. *Advances in Neural Information Processing Systems*, 31, 2018.

Mathieu Andreux, Jean Ogier du Terrail, Constance Beguier, and Eric W Tramel. Siloed federated learning for multi-centric histopathology datasets. In *Domain Adaptation and Representation Transfer, and Distributed and Collaborative Learning*, pp. 129–139. Springer, 2020a.

Mathieu Andreux, Andre Manoel, Romuald Menuet, Charlie Saillard, and Chloé Simpson. Federated survival analysis with discrete-time Cox models. *arXiv preprint arXiv:2006.08997*, 2020b.

Derek C Angus, Walter T Linde-Zwirble, Jeffrey Lidicker, Gilles Clermont, Joseph Carcillo, and Michael R Pinsky. Epidemiology of severe sepsis in the united states: analysis of incidence, outcome, and associated costs of care. *Critical Care Medicine*, 29(7):1303–1310, 2001.

Eillie Anzilotti. Visualizing the state of global internet connectivity, Aug 2016. URL https://www.citylab.com/life/2016/08/visualizing-the-state-of-global-internet-connectivity/496328/.

Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

Abhijeet Awasthi, Sabyasachi Ghosh, Rasna Goyal, and Sunita Sarawagi. Learning from rules generalizing labeled exemplars. In *International Conference on Learning Representations*, 2020.

Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 2938–2948. PMLR, 2020.

José Luis Corcuera Bárcena, Mattia Daole, Pietro Ducange, Francesco Marcelloni, Alessandro Renda, Fabrizio Ruffini, and Alessio Schiavo. Fed-xai: Federated learning of explainable artificial intelligence models. *Italian Workshop on Explainable Artificial Intelligence*, 2022.

Shai Ben-David and Reba Schuller. Exploiting task relatedness for multiple task learning. In *Learning Theory and Kernel Machines: 16th Annual Conference on Learning Theory and 7th Kernel Workshop, COLT/Kernel 2003, Washington, DC, USA, August 24-27, 2003. Proceedings*, pp. 567–580. Springer, 2003.

Whitney Blair Berta and Ross Baker. Factors that impact the transfer and retention of best practices for reducing error in hospitals. *Health Care Management Review*, 29(2):90–97, 2004.

Daniel J Beutel, Taner Topal, Akhil Mathur, Xinchi Qiu, Javier Fernandez-Marques, Yan Gao, Lorenzo Sani, Kwing Hei Li, Titouan Parcollet, Pedro Porto Buarque de Gusmão, et al. Flower: A friendly federated learning framework. *hal-03601230*, 2022.

Avrim Blum, Nika Haghtalab, Ariel D Procaccia, and Mingda Qiao. Collaborative pac learning. *Advances in Neural Information Processing Systems*, 30, 2017.

Benedikt Boecking, Willie Neiswanger, Eric Xing, and Artur Dubrawski. Interactive weak supervision: Learning useful heuristics for data labeling. In *International Conference on Learning Representations*, 2020.

Kallista Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1175–1191, 2017.

Kallista Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloé M Kiddon, Jakub Konečný, Stefano Mazzocchi, Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roselander. Towards federated learning at scale: System design. In *Conference on Systems and Machine Learning*, 2019.

Kallista Bonawitz, Peter Kairouz, Brendan Mcmahan, and Daniel Ramage. Federated learning and privacy. *Communications of the ACM*, 65(4):90–97, 2022.

Alexander Borzunov, Dmitry Baranchuk, Tim Dettmers, Max Ryabinin, Younes Belkada, Artem Chumachenko, Pavel Samygin, and Colin Raffel. Petals: Collaborative inference and fine-tuning of large models. *arXiv preprint arXiv:2209.01188*, 2022.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs. *Software*, 2018. URL http://github.com/google/jax.

Eoin Brophy, Maarten De Vos, Geraldine Boylan, and Tomas Ward. Estimation of continuous blood pressure from ppg via a federated learning approach. *Sensors*, 21(18):6311, 2021.

Peggy Bui and Yuan Liu. Using ai to help find answers to common skin conditions. *online*, 2021. URL https://blog.google/technology/health/ai-dermatology-preview-io-2021/.

Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečnỳ, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. LEAF: A benchmark for federated settings. *Workshop on Federated Learning for Data Privacy and Confidentiality at NeurIPS*, 2019a. URL https://arxiv.org/abs/1812.01097.

Sebastian Caldas, Jakub Konečnỳ, H Brendan McMahan, and Ameet Talwalkar. Expanding the reach of federated learning by reducing client resource requirements. *Workshop on Federated Learning for Data Privacy and Confidentiality at NeurIPS*, 2019b. URL https://arxiv.org/abs/1812.07210.

Sebastian Caldas, Vincent Jeanselme, Gilles Clermont, Michael R. Pinsky, and Artur Dubrawski. A case for federated learning: Enabling and leveraging inter-hospital collaboration. In *American Thoracic Society International Conference*. American Thoracic Society, 2020.

Sebastian Caldas, Jieshi Chen, and Artur Dubrawski. Using machine learning to support transfer of best practices in healthcare. In *AMIA Annual Symposium Proceedings*, volume 2021, pp. 265. American Medical Informatics Association, 2021a. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8861698/.

Sebastian Caldas, Joo Heung Yoon, Michael R. Pinsky, Gilles Clermont, and Artur Dubrawski. Understanding clinical collaborations through federated classifier selection. In Ken Jung, Serena Yeung, Mark Sendak, Michael Sjoding, and Rajesh Ranganath (eds.), *Proceedings of the 6th Machine Learning for Healthcare Conference*, volume 149 of *Proceedings of Machine Learning Research*, pp. 126–145. PMLR, 06–07 Aug 2021b. URL https://proceedings.mlr.press/v149/caldas21a.html.

Sebastian Caldas, Mononito Goswami, and Artur Dubrawski. Encoding expert knowledge into federated learning using weak supervision. *ICLR 2023 workshop on Machine Learning for IoT: Datasets, Perception, and Understanding*, 2023.

Emmanuel J Candes, Justin K Romberg, and Terence Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 59(8):1207–1223, 2006.

Fei Chen, Zhenhua Dong, Zhenguo Li, and Xiuqiang He. Federated meta-learning for recommendation. *arXiv preprint arXiv:1802.07876*, 2018.

Mayee Chen, Benjamin Cohen-Wang, Stephen Mussmann, Frederic Sala, and Christopher Ré. Comparing the value of labeled and unlabeled data in method-of-moments latent variable estimation. In *International Conference on Artificial Intelligence and Statistics*, pp. 3286–3294. PMLR, 2021.

Olivia Choudhury, Aris Gkoulalas-Divanis, Theodoros Salonidis, Issa Sylla, Yoonyoung Park, Grace Hsu, and Amar Das. Differential privacy-enabled federated learning for sensitive health data. *CoRR*, abs/1910.02578, 2019. URL http://arxiv.org/abs/1910.02578.

Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. EMNIST: Extending MNIST to handwritten letters. In *2017 international joint conference on neural networks (IJCNN)*, pp. 2921–2926. IEEE, 2017.

Rafael MO Cruz, Robert Sabourin, and George DC Cavalcanti. Dynamic classifier selection: Recent advances and perspectives. *Information Fusion*, 41:195–216, 2018.

Alicia Curth, Patrick Thoral, Wilco van den Wildenberg, Peter Bijlstra, Daan de Bruin, Paul Elbers, and Mattia Fornasa. Transferring clinical prediction models across hospitals and electronic health record systems. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 605–621. Springer, 2019.

Christopher De Sa, Megan Leszczynski, Jian Zhang, Alana Marzoev, Christopher R Aberger, Kunle Olukotun, and Christopher Ré. High-accuracy low-precision training. *arXiv preprint arXiv:1803.03383*, 2018.

Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Andrew Senior, Paul Tucker, Ke Yang, Quoc V Le, et al. Large scale distributed deep networks. In *Advances in neural information processing systems*, pp. 1223–1231, 2012.

Don Kurian Dennis, Abhishek Shetty, Anish Sevekari, Kazuhito Koishida, and Virginia Smith. Progressive knowledge distillation: Building ensembles for efficient inference. *arXiv preprint arXiv:2302.10093*, 2023.

Arnab Dey, Mononito Goswami, Joo Heung Yoon, Gilles Clermont, Michael Pinsky, Marilyn Hravnak, and Artur Dubrawski. Weakly supervised classification of vital sign alerts as real or artifact. In *AMIA Annual Symposium Proceedings*, volume 2022. American Medical Informatics Association, 2022.

Hui Ding, Goce Trajcevski, Peter Scheuermann, Xiaoyue Wang, and Eamonn Keogh. Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment*, 1(2):1542–1552, 2008.

Jie Ding, Eric Tramel, Anit Kumar Sahu, Shuang Wu, Salman Avestimehr, and Tao Zhang. Federated learning challenges and opportunities: An outlook. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8752–8756. IEEE, 2022.

Jared A Dunnmon, Alexander J Ratner, Khaled Saab, Nishith Khandwala, Matthew Markert, Hersh Sagreiya, Roger Goldman, Christopher Lee-Messer, Matthew P Lungren, Daniel L Rubin, et al. Cross-modal data programming enables rapid medical machine learning. *Patterns*, 1(2):100019, 2020.

Glyn Elwyn, Mark Taubert, and Jenny Kowalczuk. Sticky knowledge: a possible model for investigating implementation in healthcare contexts. *Implementation Science*, 2(1):1–8, 2007.

Gongfan Fang, Xinyin Ma, Mingli Song, Michael Bi Mi, and Xinchao Wang. Depgraph: Towards any structural pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16091–16101, 2023.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, 2017.

Madalina Fiterau and Artur Dubrawski. Projection retrieval for classification. In *Advances in Neural Information Processing Systems*, pp. 3023–3031, 2012.

Kathleen Kara Fitzpatrick, Alison Darcy, and Molly Vierhile. Delivering cognitive behavior therapy to young adults with symptoms of depression and anxiety using a fully automated conversational agent (woebot): A randomized controlled trial. *JMIR Ment Health*, 4(2):e19, Jun 2017. ISSN 2368-7959. doi: 10.2196/mental.7785. URL http://www.ncbi.nlm.nih.gov/pubmed/28588005.

Simon Foucart and Holger Rauhut. *A mathematical introduction to compressive sensing*, volume 1. Birkhäuser Basel, 2013.

Jason A Fries, Paroma Varma, Vincent S Chen, Ke Xiao, Heliodoro Tejeda, Priyanka Saha, Jared Dunnmon, Henry Chubb, Shiraz Maskatia, Madalina Fiterau, et al. Weakly supervised classification of aortic valve malformations using unlabeled cardiac mri sequences. *Nature communications*, 10(1):1–10, 2019.

Daniel Fu, Mayee Chen, Frederic Sala, Sarah Hooper, Kayvon Fatahalian, and Christopher Ré. Fast and three-rious: Speeding up weak supervision with triplet methods. In *International Conference on Machine Learning*, pp. 3280–3291. PMLR, 2020.

Chufan Gao, Mononito Goswami, Jieshi Chen, and Artur Dubrawski. Classifying unstructured clinical notes via automatic weak supervision. In *Proceedings of the Machine Learning for Healthcare Conference*, Proceedings of Machine Learning Research. PMLR, 2022.

Roman Garnett, Yamuna Krishnamurthy, Xuehan Xiong, Jeff Schneider, and Richard Mann. Bayesian optimal active search and surveying. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pp. 843–850, 2012.

Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé Iii, and Kate Crawford. Datasheets for datasets. *Communications of the ACM*, 64(12):86–92, 2021.

Robin C Geyer, Tassilo Klein, and Moin Nabi. Differentially private federated learning: A client level perspective. *arXiv preprint arXiv:1712.07557*, 2017.

Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. *Project Report, Stanford*, 2009.

Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *circulation*, 101(23):e215–e220, 2000.

Mononito Goswami, Benedikt Boecking, and Artur Dubrawski. Weak supervision for affordable modeling of electrocardiogram data. In *AMIA Annual Symposium Proceedings*, volume 2021, pp. 536. American Medical Informatics Association, 2021.

Dhruv Guliani, Lillian Zhou, Changwan Ryu, Tien-Ju Yang, Harry Zhang, Yonghui Xiao, Françoise Beaufays, and Giovanni Motta. Enabling on-device training of speech recognition models with federated dropout. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8757–8761, 2022. doi: 10.1109/ICASSP43922.2022.9746226.

Gustavo Guzman, Janna Anneke Fitzgerald, Liz Fulop, Kathryn Hayes, Arthur Poropat, Mark Avery, Steve Campbell, Ron Fisher, Rod Gapp, Carmel Herington, et al. How best practices are copied, transferred, or translated between health care facilities: a conceptual framework. *Health care management review*, 40(3):193–202, 2015.

Farzin Haddadpour, Mohammad Mahdi Kamani, Aryan Mokhtari, and Mehrdad Mahdavi. Federated learning with compression: Unified analysis and sharp guarantees. In *International Conference on Artificial Intelligence and Statistics*, pp. 2350–2358. PMLR, 2021.

Nika Haghtalab, Michael Jordan, and Eric Zhao. On-demand sampling: Learning optimally from multiple distributions. *Advances in Neural Information Processing Systems*, 35:406–419, 2022.

Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *International Conference on Learning Representations*, 2016.

Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*, 2018.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015.

Samuel Horvath, Stefanos Laskaridis, Mario Almeida, Ilias Leontiadis, Stylianos Venieris, and Nicholas Lane. Fjord: Fair and accurate federated learning under heterogeneous targets with ordered dropout. *Advances in Neural Information Processing Systems*, 34:12876–12889, 2021.

Samuel Horvóth, Chen-Yu Ho, Ludovit Horvath, Atal Narayan Sahu, Marco Canini, and Peter Richtárik. Natural compression for distributed deep learning. In *Mathematical and Scientific Machine Learning*, pp. 129–141. PMLR, 2022.

Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019.

Li Huang, Andrew L Shea, Huining Qian, Aditya Masurkar, Hao Deng, and Dianbo Liu. Patient clustering improves efficiency of federated machine learning to predict mortality and hospital stay time using distributed electronic medical records. *Journal of biomedical informatics*, 99: 103291, 2019.

Truong Thu Huong, Ta Phuong Bac, Dao Minh Long, Tran Duc Luong, Nguyen Minh Dan, Bui Doan Thang, Kim Phuc Tran, et al. Detecting cyberattacks using anomaly detection in industrial control systems: A federated learning approach. *Computers in Industry*, 132:103509, 2021.

Wonyong Jeong, Jaehong Yoon, Eunho Yang, and Sung Ju Hwang. Federated semi-supervised learning with inter-client consistency & disjoint learning. In *International Conference on Learning Representations*, 2020.

Shali Jiang, Gustavo Malkomes, Geoff Converse, Alyssa Shofner, Benjamin Moseley, and Roman Garnett. Efficient nonmyopic active search. In *International Conference on Machine Learning*, pp. 1714–1723. PMLR, 2017.

Yihan Jiang, Jakub Konečnỳ, Keith Rush, and Sreeram Kannan. Improving federated learning personalization via model agnostic meta learning. *arXiv preprint arXiv:1909.12488*, 2019.

Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Hubert Eichner, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaid Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konecný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Hang Qi, Daniel Ramage, Ramesh Raskar, Mariana Raykova, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021. ISSN 1935-8237. doi: 10.1561/2200000083. URL http://dx.doi.org/10.1561/2200000083.

Michael Kamp, Linara Adilova, Joachim Sicking, Fabian Hüger, Peter Schlicht, Tim Wirtz, and Stefan Wrobel. Efficient decentralized deep learning by dynamic model averaging. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2018, Dublin, Ireland, September 10–14, 2018, Proceedings, Part I 18*, pp. 393–409. Springer, 2019.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pp. 5132–5143. PMLR, 2020.

Boris Sergeevich Kashin. Diameters of some finite-dimensional sets and classes of smooth functions. *Izvestiya Rossiiskoi Akademii Nauk. Seriya Matematicheskaya*, 41(2):334–351, 1977.

Mikhail Khodak, Maria-Florina F Balcan, and Ameet S Talwalkar. Adaptive gradient-based meta-learning methods. *Advances in Neural Information Processing Systems*, 32, 2019a.

Mikhail Khodak, Maria Florina-Balcan, and Ameet Talwalkar. Adaptive gradient-based meta-learning methods. *Advances in Neural Information Processing Systems*, 2019b.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Jakub Konečný and Peter Richtárik. Randomized distributed mean estimation: Accuracy vs communication. *arXiv preprint arXiv:1611.07555*, 2016.

Jakub Konečný, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Abhishek Kumar and Hal Daumé III. Learning task grouping and overlap in multi-task learning. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pp. 1723–1730, 2012.

Ilja Kuzborskij and Francesco Orabona. Stability and hypothesis transfer learning. In *International Conference on Machine Learning*, pp. 942–950. PMLR, 2013.

Yassine Laguel, Krishna Pillutla, Jérôme Malick, and Zaid Harchaoui. A superquantile approach to federated learning with heterogeneous devices. In *55th Annual Conference on Information Sciences and Systems, CISS 2021, Baltimore, MD, USA, March 24-26, 2021*, pp. 1–6. IEEE, 2021.

Brenden Lake, Ruslan Salakhutdinov, Jason Gross, and Joshua Tenenbaum. One shot learning of simple visual concepts. In *Annual Meeting of the Cognitive Science Society*, 2011.

Yann LeCun. The MNIST database of handwritten digits. *http://yann.lecun.com/exdb/mnist/*, 1998.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Giwoong Lee, Eunho Yang, and Sung Hwang. Asymmetric multi-task learning based on task relatedness and loss. In *International Conference on Machine Learning*, 2016.

Gyemin Lee, Ilan Rubinfeld, and Zeeshan Syed. Adapting surgical models to individual hospitals using transfer learning. In *2012 IEEE 12th international conference on data mining workshops*, pp. 57–63. IEEE, 2012.

David Leroy, Alice Coucke, Thibaut Lavril, Thibault Gisselbrecht, and Joseph Dureau. Federated learning for keyword spotting. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2019.

Jeffrey Li, Mikhail Khodak, Sebastian Caldas, and Ameet Talwalkar. Differentially private meta-learning. In *International Conference on Learning Representations*, 2019a.

Jiacheng Li, Haibo Ding, Jingbo Shang, Julian McAuley, and Zhe Feng. Weakly supervised named entity tagging with learnable logical rules. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 4568–4581, 2021.

Jingtao Li, Lingjuan Lyu, Daisuke Iso, Chaitali Chakrabarti, and Michael Spranger. Mocosfl: enabling cross-client collaborative self-supervised learning. In *Workshop on Federated Learning: Recent Advances and New Challenges (in Conjunction with NeurIPS 2022)*, 2022.

Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020a.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In I. Dhillon, D. Papailiopoulos, and V. Sze (eds.), *Proceedings of Machine Learning and Systems*, volume 2, pp. 429–450, 2020b.

Tian Li, Maziar Sanjabi, Ahmad Beirami, and Virginia Smith. Fair resource allocation in federated learning. In *International Conference on Learning Representations*, 2020c.

Wenqi Li, Fausto Milletarì, Daguang Xu, Nicola Rieke, Jonny Hancox, Wentao Zhu, Maximilian Baust, Yan Cheng, Sébastien Ourselin, M Jorge Cardoso, et al. Privacy-preserving federated brain tumour segmentation. In *International Workshop on Machine Learning in Medical Imaging*, pp. 133–141. Springer, 2019b.

Quande Liu, Hongzheng Yang, Qi Dou, and Pheng-Ann Heng. Federated semi-supervised medical image classification via inter-client relation matching. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 325–335. Springer, 2021.

Yi Liu, Sahil Garg, Jiangtian Nie, Yang Zhang, Zehui Xiong, Jiawen Kang, and M Shamim Hossain. Deep anomaly detection for time-series data in industrial iot: A communication-efficient on-device federated learning approach. *IEEE Internet of Things Journal*, 8(8):6348–6358, 2020.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *International Conference on Computer Vision*, 2015.

Nan Lu, Zhao Wang, Xiaoxiao Li, Gang Niu, Qi Dou, and Masashi Sugiyama. Federated learning from only unlabeled data with class-conditional-sharing clients. In *International Conference on Learning Representations*, 2021.

Yurii Lyubarskii and Roman Vershynin. Uncertainty principles and vector quantization. *IEEE Transactions on Information Theory*, 56(7):3491–3501, 2010.

Ahmed M Abdelmoniem, Ahmed Elzanaty, Mohamed-Slim Alouini, and Marco Canini. An efficient statistical-based gradient compression technique for distributed training systems. *Proceedings of Machine Learning and Systems*, 3:297–322, 2021.

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pp. 142–150, 2011.

Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation with multiple sources. *Advances in neural information processing systems*, 21, 2008.

Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation: Learning bounds and algorithms. *arXiv preprint arXiv:0902.3430*, 2009.

Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619*, 2020.

Chiara Marzi, Marco Giannelli, Andrea Barucci, Carlo Tessa, Mario Mascalchi, and Stefano Diciotti. Efficacy of mri data harmonization in the age of machine learning. a multicenter study across 36 datasets. *arXiv preprint arXiv:2211.04125*, 2022.

Michael Matheny, S Thadaney Israni, Mahnoor Ahmed, and Danielle Whicher. Artificial intelligence in health care: The hope, the hype, the promise, the peril. *NAM Special Publication. Washington, DC: National Academy of Medicine*, pp. 154, 2019.

Scott Mayer McKinney, Marcin Sieniek, Varun Godbole, Jonathan Godwin, Natasha Antropova, Hutan Ashrafian, Trevor Back, Mary Chesus, Greg S Corrado, Ara Darzi, et al. International evaluation of an ai system for breast cancer screening. *Nature*, 577(7788):89–94, 2020.

Brendan McMahan and Abhradeep Thakurta. Federated Learning with Formal Differential Privacy Guarantees, February 2022. URL https://ai.googleblog.com/2022/02/federated-learning-with-formal.html.

H Brendan McMahan and Daniel Ramage. Federated learning: Collaborative machine learning without centralized training data. *online*, 2017. URL https://ai.googleblog.com/2017/04/federated-learning-collaborative.html.

H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, 2017.

H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *International Conference on Learning Representations*, 2018.

Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Inference attacks against collaborative learning. *arXiv preprint arXiv:1805.04049*, 2018.

Meta AI Research. Federated Learning Simulator (FLSim), 2022. URL https://github.com/facebookresearch/FLSim/.

Sarthak Mittal, Yoshua Bengio, and Guillaume Lajoie. Is a modular architecture enough? In *Advances in Neural Information Processing Systems*, 2022.

Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. Agnostic federated learning. In *International Conference on Machine Learning*, pp. 4615–4625. PMLR, 2019.

George B Moody and Roger G Mark. The impact of the mit-bih arrhythmia database. *IEEE engineering in medicine and biology magazine*, 20(3):45–50, 2001.

Andrew Moore and Jeff Schneider. Real-valued all-dimensions search: Low-overhead rapid searching over subsets of attributes. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*, pp. 360–369. Morgan Kaufmann Publishers Inc., 2002.

Abdullah Mueen, Eamonn Keogh, Qiang Zhu, Sydney Cash, and Brandon Westover. Exact discovery of time series motifs. In *Proceedings of the 2009 SIAM international conference on data mining*, pp. 473–484. SIAM, 2009.

Keerthiram Murugesan and Jaime Carbonell. Multi-task multiple kernel relationship learning. In *SIAM International Conference on Data Mining*, 2017.

Basil Mustafa, Aaron Loh, Jan Freyberg, Patricia MacWilliams, Alan Karthikesalingam, Neil Houlsby, and Vivek Natarajan. Supervised transfer learning at scale for medical imaging. *arXiv preprint arXiv:2101.05913*, 2021.

Yang Nan, Javier Del Ser, Simon Walsh, Carola Schönlieb, Michael Roberts, Ian Selby, Kit Howard, John Owen, Jon Neville, Julien Guiot, et al. Data harmonisation for information fusion in digital healthcare: A state-of-the-art systematic review, meta-analysis and future research directions. *Information Fusion*, 82:99–122, 2022.

Mona Nashaat, Aindrila Ghosh, James Miller, and Shaikh Quader. Asterisk: Generating large training datasets with automatic active supervision. *ACM Transactions on Data Science*, 1(2): 1–25, 2020.

Shamim Nemati, Andre Holder, Fereshteh Razmi, Matthew D Stanley, Gari D Clifford, and Timothy G Buchman. An interpretable machine learning model for accurate prediction of sepsis in the ICU. *Critical care medicine*, 46(4):547, 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)*, pp. 188–197, 2019.

Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

Jean Ogier du Terrail, Samy-Safwan Ayed, Edwige Cyffers, Felix Grimberg, Chaoyang He, Regis Loeb, Paul Mangold, Tanguy Marchand, Othmane Marfoq, Erum Mushtaq, et al. Flamby: Datasets and benchmarks for cross-silo federated learning in realistic healthcare settings. *Advances in Neural Information Processing Systems*, 35:5315–5334, 2022.

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 32*, pp. 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

Matthias Paulik, Matt Seigel, Henry Mason, Dominic Telaar, Joris Kluivers, Rogier van Dalen, Chi Wai Lau, Luke Carlson, Filip Granqvist, Chris Vandevelde, et al. Federated evaluation and tuning for on-device personalization: System design & applications. *arXiv preprint arXiv:2102.08503*, 2021.

Matthias Perleth, Elke Jakubowski, and Reinhard Busse. What is 'best practice'in health care? state of the art and perspectives in improving the effectiveness and efficiency of the european health care systems. *Health policy*, 56(3):235–250, 2001.

Pew Research Center. Mobile fact sheet, Feb 2018. URL http://www.pewinternet.org/fact-sheet/mobile/.

Krishna Pillutla, Sham M. Kakade, and Zaid Harchaoui. Robust Aggregation for Federated Learning. *IEEE Transactions on Signal Processing*, 70:1142–1154, 2022. doi: 10.1109/TSP.2022.3153135.

Barnabás Póczos and Jeff Schneider. On the estimation of alpha-divergences. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 609–617. JMLR Workshop and Conference Proceedings, 2011.

Willa Potosnak. Robust rule learning for reliable and interpretable insight into expertise transfer opportunities. In *Proceedings of the Thirty-Sixth AAAI Conference on Artificial Intelligence*. Research Summary, 2022.

Willa Potosnak, Sebastian Caldas, Keith. A. Dufendach, Gilles Clermont, Kyle Miller, and Artur Dubrawski. Robust interpretable rule learning to identify expertise transfer opportunities in healthcare. In *Bridging the Gap: From Machine Learning Research to Clinical Practice*. NeurIPS Workshop, 2021.

Colin Raffel. Building machine learning models like open source software. *Communications of the ACM*, 66(2):38–40, 2023.

Maithra Raghu, Chiyuan Zhang, Jon Kleinberg, and Samy Bengio. Transfusion: Understanding transfer learning for medical imaging. *Advances in neural information processing systems*, 32, 2019.

Swaroop Ramaswamy, Rajiv Mathews, Kanishka Rao, and Françoise Beaufays. Federated learning for emoji prediction in a mobile keyboard. *arXiv preprint arXiv:1906.04329*, 2019.

Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. Snorkel: Rapid training data creation with weak supervision. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, volume 11, pp. 269. NIH Public Access, 2017.

Alexander Ratner, Braden Hancock, Jared Dunnmon, Frederic Sala, Shreyash Pandey, and Christopher Ré. Training complex models with multi-task weak supervision. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 4763–4771, 2019.

Alexander J Ratner, Christopher M De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. Data programming: Creating large training sets, quickly. *Advances in neural information processing systems*, 29, 2016.

Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. *https://openreview.net/pdf?id=rJY0-Kcll*, 2016.

Sashank J Reddi, Jakub Konečný, Peter Richtárik, Barnabás Póczós, and Alex Smola. Aide: fast and communication efficient distributed optimization. *arXiv preprint arXiv:1608.06879*, 2016.

Sashank J Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečnỳ, Sanjiv Kumar, and Hugh Brendan McMahan. Adaptive federated optimization. In *International Conference on Learning Representations*, 2020.

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 https://arxiv.org/abs/1908.10084.

Matthew A Reyna, Chris Josef, Salman Seyedi, Russell Jeter, Supreeth P Shashikumar, M Brandon Westover, Ashish Sharma, Shamim Nemati, and Gari D Clifford. Early prediction of sepsis from clinical data: the physionet/computing in cardiology challenge 2019. In *2019 Computing in Cardiology (CinC)*, pp. Page–1. IEEE, 2019.

Nicola Rieke, Jonny Hancox, Wenqi Li, Fausto Milletari, Holger R Roth, Shadi Albarqouni, Spyridon Bakas, Mathieu N Galtier, Bennett A Landman, Klaus Maier-Hein, et al. The future of digital health with federated learning. *NPJ digital medicine*, 3(1):1–7, 2020.

Clemens Rosenbaum, Ignacio Cases, Matthew Riemer, and Tim Klinger. Routing networks and the challenges of modular and compositional computation. *arXiv preprint arXiv:1904.12774*, 2019.

Salva Rühling Cachay, Benedikt Boecking, and Artur Dubrawski. End-to-end weak supervision. *Advances in Neural Information Processing Systems*, 34, 2021.

Adam Sadilek, Luyang Liu, Dung Nguyen, Methun Kamruzzaman, Stylianos Serghiou, Benjamin Rader, Alex Ingerman, Stefan Mellem, Peter Kairouz, Elaine O Nsoesie, et al. Privacy-first health research with federated learning. *NPJ digital medicine*, 4(1):1–8, 2021.

Sentence Transformers. Hugging Face Sentence Transformers. https://huggingface.co/sentence-transformers/all-mpnet-base-v2, 2019. Accessed: 2023-06-07.

Micah J Sheller, G Anthony Reina, Brandon Edwards, Jason Martin, and Spyridon Bakas. Multi-institutional deep learning modeling without sharing patient data: A feasibility study on brain tumor segmentation. In *International MICCAI Brainlesion Workshop*, pp. 92–104. Springer, 2018.

Martin Rozycki Shukla et al. Saima rathore, spyridon bakas, hamed akbari, gaurav. In *Proc. of SPIE Vol*, volume 10575, pp. 1057509–1, 2018.

George CM Siontis, Ioanna Tzoulaki, Peter J Castaldi, and John PA Ioannidis. External validation of new risk prediction models is infrequent and reveals worse prognostic discrimination. *Journal of clinical epidemiology*, 68(1):25–34, 2015.

Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet Talwalkar. Federated multi-task learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 4427–4437, 2017.

Virginia Smith, Simone Forte, Ma Chenxin, Martin Takáč, Michael I Jordan, and Martin Jaggi. Cocoa: A general framework for communication-efficient distributed optimization. *Journal of Machine Learning Research*, 18:230, 2018.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, 2017.

Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *International Conference on Learning Representations (workshop track)*, 2015.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

Sebastian U Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. Sparsified sgd with memory. *Advances in Neural Information Processing Systems*, 31, 2018.

Adarsh Subbaswamy, Roy Adams, and Suchi Saria. Evaluating model robustness and stability to dataset shift. In *International Conference on Artificial Intelligence and Statistics*, pp. 2611–2619. PMLR, 2021.

Ananda Theertha Suresh, X Yu Felix, Sanjiv Kumar, and H Brendan McMahan. Distributed mean estimation with limited communication. In *International conference on machine learning*, pp. 3329–3337. PMLR, 2017a.

Ananda Theertha Suresh, Felix X Yu, Sanjiv Kumar, and H Brendan McMahan. Distributed mean estimation with limited communication. In *International Conference on Machine Learning*, pp. 3329–3337, 2017b.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Gabriel Szulanski. *Sticky knowledge: Barriers to knowing in the firm*. Sage, 2002.

Hanlin Tang, Chen Yu, Xiangru Lian, Tong Zhang, and Ji Liu. Doublesqueeze: Parallel stochastic gradient descent with double-pass error-compensated compression. In *International Conference on Machine Learning*, pp. 6155–6165. PMLR, 2019.

Nicola Tonellotto, Alberto Gotta, Franco Maria Nardini, Daniele Gadler, and Fabrizio Silvestri. Neural network quantization in federated learning at the edge. *Information Sciences*, 575: 417–436, 2021.

Haridimos Tsoukas and Efi Vladimirou. What is organizational knowledge? *Journal of management studies*, 38(7):973–993, 2001.

Gregor Ulm, Emil Gustavsson, and Mats Jirstrand. Functional federated learning in erlang (ffl-erl). In *International Workshop on Functional and Constraint Logic Programming*, 2018.

Willem G Van Panhuis, Proma Paul, Claudia Emerson, John Grefenstette, Richard Wilder, Abraham J Herbst, David Heymann, and Donald S Burke. A systematic review of barriers to data sharing in public health. *BMC public health*, 14(1):1–9, 2014.

Paroma Varma and Christopher Ré. Snuba: Automating weak supervision to label training data. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, volume 12, pp. 223. NIH Public Access, 2018.

Oriol Vinyals, Charles Blundell, Tim Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, 2016.

Thijs Vogels, Sai Praneeth Karimireddy, and Martin Jaggi. Powersgd: Practical low-rank gradient compression for distributed optimization. *Advances in Neural Information Processing Systems*, 32, 2019.

Boxin Wang, Yibo Jacky Zhang, Yuan Cao, Bo Li, H Brendan McMahan, Sewoong Oh, Zheng Xu, and Manzil Zaheer. Can public large language models help private cross-device federated learning? *arXiv preprint arXiv:2305.12132*, 2023.

Hongyi Wang, Scott Sievert, Shengchao Liu, Zachary Charles, Dimitris Papailiopoulos, and Stephen Wright. Atomo: Communication-efficient learning via atomic sparsification. *Advances in Neural Information Processing Systems*, 31, 2018.

Jianyu Wang and Gauri Joshi. Cooperative sgd: A unified framework for the design and analysis of local-update sgd algorithms. *The Journal of Machine Learning Research*, 22(1):9709–9758, 2021.

Jianyu Wang, Zachary Charles, Zheng Xu, Gauri Joshi, H Brendan McMahan, Maruan Al-Shedivat, Galen Andrew, Salman Avestimehr, Katharine Daly, Deepesh Data, et al. A field guide to federated optimization. *arXiv preprint arXiv:2107.06917*, 2021.

Shiqiang Wang, Tiffany Tuor, Theodoros Salonidis, Kin K Leung, Christian Makaya, Ting He, and Kevin Chan. Adaptive federated learning in resource constrained edge computing systems. *IEEE Journal on Selected Areas in Communications*, 2019.

Yujia Wang, Lu Lin, and Jinghui Chen. Communication-efficient adaptive federated learning. In *International Conference on Machine Learning*, pp. 22802–22838. PMLR, 2022.

Jenna Wiens, John Guttag, and Eric Horvitz. A study in transfer learning: leveraging data from multiple hospitals to enhance hospital-specific predictions. *Journal of the American Medical Informatics Association*, 21(4):699–706, 2014.

William Shakespeare. The Complete Works of William Shakespeare. Publicly available at //www.gutenberg.org/ebooks/100, n.d.

Elena Williams, Manuel Kienast, Evelyn Medawar, Janis Reinelt, Alberto Merola, Sophie Anne Ines Klopfenstein, Anne Rike Flint, Patrick Heeren, Akira-Sebastian Poncette, Felix Balzer, et al. A standardized clinical data harmonization pipeline for scalable ai application deployment (fhir-dhp): Validation and usability study. *JMIR Medical Informatics*, 11:e43847, 2023.

Jiaxiang Wu, Weidong Huang, Junzhou Huang, and Tong Zhang. Error compensated quantized sgd and its applications to large-scale distributed optimization. In *International Conference on Machine Learning*, pp. 5325–5333. PMLR, 2018.

Yawen Wu, Dewen Zeng, Zhepeng Wang, Yi Sheng, Lei Yang, Alaina J James, Yiyu Shi, and Jingtong Hu. Federated contrastive learning for dermatological disease diagnosis via on-device learning. In *2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, pp. 1–7. IEEE, 2021.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pp. 38087–38099. PMLR, 2023.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit bayesian inference. In *International Conference on Learning Representations*, 2021.

Xiaohang Xu, Hao Peng, Lichao Sun, Md Zakirul Alam Bhuiyan, Lianzhong Liu, and Lifang He. Fedmood: Federated learning on mobile health data for mood detection. *arXiv preprint arXiv:2102.09342*, 2021.

Ya Xue, Xuejun Liao, Lawrence Carin, and Balaji Krishnapuram. Multi-task learning for classification with dirichlet process priors. *Journal of Machine Learning Research*, 8(Jan):35–63, 2007.

Timothy Yang, Galen Andrew, Hubert Eichner, Haicheng Sun, Wei Li, Nicholas Kong, Daniel Ramage, and Françoise Beaufays. Applied federated learning: Improving google keyboard query suggestions. *arXiv preprint arXiv:1812.02903*, 2018.

Binhang Yuan, Yongjun He, Jared Davis, Tianyi Zhang, Tri Dao, Beidi Chen, Percy S Liang, Christopher Re, and Ce Zhang. Decentralized training of foundation models in heterogeneous environments. *Advances in Neural Information Processing Systems*, 35:25464–25477, 2022.

John R Zech, Marcus A Badgeley, Manway Liu, Anthony B Costa, Joseph J Titano, and Eric Karl Oermann. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: a cross-sectional study. *PLoS medicine*, 15(11):e1002683, 2018.

Jieyu Zhang, Cheng-Yu Hsieh, Yue Yu, Chao Zhang, and Alexander Ratner. A survey on programmatic weak supervision. *arXiv preprint arXiv:2202.05433*, 2022.

Mufeng Zhang, Yining Wang, and Tao Luo. Federated learning for arrhythmia detection of non-iid ecg. In *2020 IEEE 6th International Conference on Computer and Communications (ICCC)*, pp. 1176–1180. IEEE, 2020.

Yi Zhang and Jeff G Schneider. Learning multiple tasks with a sparse matrix-normal penalty. In *Advances in Neural Information Processing Systems*, 2010.

Fanglan Zheng, Kun Li, Jiang Tian, Xiaojia Xiang, et al. A vertical federated learning method for interpretable scorecard and its application in credit scoring. *arXiv preprint arXiv:2009.06218*, 2020.

Shuai Zheng, Ziyue Huang, and James Kwok. Communication-efficient distributed blockwise momentum sgd with error-feedback. *Advances in Neural Information Processing Systems*, 32, 2019.

Weiming Zhuang, Xin Gan, Yonggang Wen, Shuai Zhang, and Shuai Yi. Collaborative unsupervised visual representation learning from decentralized data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4912–4921, 2021.