Applied Mathematics of the Future

Kin G. Olivares

May 2023

CMU-ML-23-102

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

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

This work was partially supported by the Defense Advanced Research Projects Agency (award FA8750-17-2-0130), the National Science Foundation (grant 2038612), the Space Technology Research Institutes grant from NASA's Space Technology Research Grants Program, the U.S. Department of Homeland Security (award 18DN-ARI-00031) and the International Institute of Forecasters SAS award.

Keywords: machine learning, deep learning, interpretability, time series decomposition, long-horizon forecasting, wavelets, time frequency analysis, hierarchical forecasting, probabilistic coherence, multivariate mixtures,

This thesis is dedicated to Stefania La Vattiata and to the memory of Jorge Olivares.

Abstract

Novel learning algorithms have enhanced our ability to acquire knowledge solely from past observations of single events to learn from the observations of several related events. This ability to leverage shared useful information across time series is causing a paradigm shift in the time-series forecasting practice. Still, machine learning-based forecasting faces pressing challenges that limit its usability, usefulness, and attainable real-world impact, including human interpretability, the ability to leverage structured information, generalization capabilities, and computational costs. This thesis tackles these challenges by bridging the gap between machine learning and classic statistical forecasting methods.

We organized the thesis as follows. We introduce the time-series forecasting task, accompanied by a short review of modern forecasting models, their optimization, and forecast evaluation methods. In the following chapters, we present our approach with three case studies. First, we augment state-of-the-art neural forecasting algorithms with interpretability capabilities inspired by time series decomposition analysis; we illustrate its application in the short-term electricity price forecasting task. Second, we improve neural forecasting generalization and computational efficiency in the long-horizon setting through a novel wavelet-inspired algorithm that assembles its predictions sequentially, emphasizing components with different frequencies and scales. Third, we tackle the hierarchical forecasting task, a regression problem with linear aggregation constraints, by augmenting neural forecasting architectures with a specialized probability mixture capable of incorporating the aggregation constraints in its construction. Our approach improves upon the current state-of-the-art in each of the considered domains.

Acknowledgments

I am fortunate and honored to have had so many outstanding individuals supporting me throughout my journey, without whom the completion of this work would have been impossible.

I am incredibly grateful to my advisor, Artur Dubrawski, for his unwavering belief in my abilities and help to overcome the challenges I faced during my research. His encouragement helped me turn my background's initial apparent limitations into my strengths and a source of inspiration for my contributions to the machine learning field. I am impressed by Artur's dedication to fostering a unique and fertile research environment in the Auton Lab.

I am grateful to my thesis committee: Barnabas Poczos, Russ Salakhutdinov, and Robert A. Stine. My interactions with them led to many ideas and intuitions behind the thesis work. Many thanks to Robert, my work at Amazon Forecasting Science team provided invaluable insight into the future of the forecasting practice. I was honored to have attended Barnabas and Russ's lectures, as their expertise played an essential role in shaping my approach; it's a privilege to have learned from you.

I sincerely thank Nganba Meetei, Ruijun Ma, Rohan Reddy, and Mengfei Cao for their support and guidance throughout my internships at Amazon Forecasting Science. They are exceptional hosts and wonderful colleagues with an unparalleled commitment to the excellence of their work. Thanks also to Gonzalo Robles, who, many years ago, sparked my curiosity about semi-parametric statistical methods.

Special thanks to my close friends and intellectual sparring partners, Max Mergenthaler and Cristian Challu, for their support and inspiration. Their presence in my life has been transformative, and our shared quest for knowledge has led to significant personal and professional growth. I also acknowledge the contributions of Federico Garza, whose efforts have enabled our work to reach a global audience and aid practitioners in rediscovering the power of classical forecasting methods. It is truly humbling to see our projects' widespread impact, which a mere year ago would have seemed unimaginable.

In addition, I want to express my sincere appreciation to my classmates and cohort members, David Luo, Kartik Gupta, Sebastian Caldas, Leqi Liu, Biswajit Paria, Nicholay Topin, Gregory Plumb, Willa Potosnak, Darby Losey, and Hillary Wehry. Time is not enough to spend with such admirable people.

Last but certainly not least, I want to express my profound gratitude to my beloved family, including my mother Nicté, my brother Inti, my grandmother Maru, my uncles Zazil, Luis, and Jorge Adrián, and my wife Stefania, whose love I treasure most in the world.

Contents

List of Figures

List of Tables

List of Algorithms

List of Theorems

Abbreviations

ADAM Adaptive Moment Estimation SGD. 17, 46, 69, 84

ADIDA Aggregate-Disaggregate Intermitent Demand Approach. 24

AR Autorregresive Models. 21, 28, 48

ARCH Autoregressive Conditional Heteroskedasticity. 22

ARIMA Autorregresive Integrated Moving Average. 18, 20, 21, 39, 71, 84

CES Complex Exponential Smoothing. 20

CNN Convolutional Neural Networks. 30

Croston Croston Method. 24

CRPS Continuous Ranked Probability Score. 12, 68

DeepAR Deep Auto Regressive Network. 31

DM Diebold-Mariano test. 13, 45

ERM Empirical Risk Minimization. 15, 25, 46, 84

ES Exponential Smoothing. 20, 21

ESRNN Exponential Smoothing Recurrent Neural Network. 31, 33, 48

ETS Error, Trend, Seasonality Exponential Smoothing State Space. 18, 20

GARCH Generalized Autoregressive Conditional Heteroskedasticity. 22

GRU Gated Recurrent Unit. 29

GW Giacomini-White test. 13, 45

HA Historic Average. 20

HYPEROPT Sequential Model-Based Hyperparameter Optimization. 18, 46, 69, 83

IMAPA Intermittent Multiple Aggregation Prediction Algorithm. 24

KNN K-Nearest Neighbor Model. 25

LSTM	Long Short Term Memory. 29
MA	Moving Average. 20, 21
MAE	Mean Absolute Error. 11, 13, 45, 83
MAP	Maximum a Posteriori Estimation. 16
MAPE	Mean Absolute Percentage Error. 11
MLE	Maximum Likelihood Estimation. 16, 21, 60
MLP	Multi Layer Perceptron. 28, 36, 63, 77
MQForecaster	Multi Quantile Forecaster Family. 31, 53, 62, 63
MQL	Multi Quantile Loss. 12, 68
MS	Markov Regime-Switching. 23
MSE	Mean Squared Error. 11, 15, 21, 83
Naive	Naive. 20, 45, 62, 68
NBEATS	Neural Basis Expansion Analysis. 31, 33, 48, 77
Prophet	Facebook Prophet. 23
QL	Quantile Loss. 12, 15
relMAE	relative Mean Absolute Error. 11, 45
relMSE	relative Mean Squared Error. 11, 68
RMSE	Root Mean Squared Error. 11, 45
RNN	Recurrent Neural Network. 29, 84
Seq2Seq	Sequence to Sequence Architecture. 31, 63
sMAPE	symmetric Mean Absolute Percentage Error. 11, 45
SNaive	Seasonal Naive. 20, 62, 71
STL	Seasonal-Trend decomposition using LOESS. 23, 39
STR	Seasonal-Trend regression. 23, 39
SVM	Support Vector Machine. 25
TCN	Temporal Convolution Network. 30, 40, 63
TFT	Temporal Fusion Transformer. 31
Transformer	Transformer. 30, 84
VAR	Vector Autoregression. 24, 57
ZIP	Zero Inflated Poisson Regression. 24, 107

Chapter 1

Introduction

1.1 Problem Statement

The time series forecasting problem concerns many areas ranging from finance and economics to healthcare analytics. As data generation increases, the forecasting necessities have evolved from the need to predict small groups of time series to predicting thousands or even millions of them. Extracting statistical patterns from the data that generalize has been the most reliable method of producing predictions. This is why machine learning has become one of the most successful approaches for the task.

Large dataset environments have witnessed Deep Learning (LeCun et al., 2015) grow in popularity as it becomes a valuable and general-purpose forecasting technique, as shown by its success in recent forecasting competitions (Makridakis et al., 2020a; Makridakis et al., 2021) where it redefined the state-of-the-art. Its benefits include:

1. **Forecasting Accuracy**: A global model is fitted simultaneously to the historical data of related time series, allowing it to share information across them; this helps to train highly parameterized and flexible models that often translates in more accurate forecasts, this technique is known as *cross learning* (Makridakis et al., 2020a). The model is able to provide forecasts for items that have little to no history available, in contrast to classical methods.
2. **Forecasting Pipeline's Simplification**: The deep learning framework is able to automatize the featurization of the dataset, while its representations exhibit longer memory. The use of global models greatly simplify the data pipelines and make the process more efficient. While the training times are bigger than other methods, deep learning techniques compensates for it during the data featurization process, which is usually extremely fast.

Plenty of methods and ideas have been tried in forecasting, with varying degrees of success. Different algorithms have strengths and weaknesses, varying complexity, development opportunities, and challenges. Machine Learning has a great potential to enhance forecasting systems, yet some limitations hinder its adoption, among them we identified the lack of interpretability, its computational scalability when dealing with large amounts of data, or long-horizon predictions.

Prompted by the interpretability and computational cost limitations of machine learning forecasting systems, in this thesis, we guide our work by the following question:

> *Can econometrics and statistical innovations be combined to advance usability, usefulness, and real-world impact of machine learning-based forecasting?*

1.2 Goal

There is often an imagined divide between statistical and econometric forecasting models and the machine learning approach. On one side, statistical models are highly determined by assumptions to model relationships between variables. These models strive to understand the underlying data-generating processes and are typically inspired by theory. On the other hand, machine learning forecasting models are characterized by their ability to assemble predictions primarily based on the data without being explicitly programmed to do so. Due to this, these models have minimal assumptions and are exceptionally flexible.

Thesis statement. My thesis aims to bridge the gap between econometric, statistical, and machine learning forecasting methods, and centers around the idea that:

> *Confining machine learning-based forecasting methods with econometric and statistical domain knowledge is necessary to improve their accuracy, interpretability, and efficiency.*

1.3 Contributions

The thesis main body presents case studies showcasing the successful application of our approach, which enhances neural forecasting methods with econometric and statistical inspirations. Below is an executive summary of the thesis' contributions.

- We introduced NBEATSx, a neural forecasting solution that extends the neural basis expansion analysis incorporating exogenous variables. NBEATSx improves accuracy through the integration of multiple information sources. The architecture also provides an interpretable signal decomposition, allowing users to visualize the impact of trend and seasonal components and their interactions with exogenous factors.
- We introduced NHITS, a neural hierarchical interpolation for time series inspired by Wavelet analysis. NHITS improves long-horizon forecasting accuracy and reduces the computation time of the multi-step forecasting strategy.
- We tackled the hierarchical forecasting challenge by combining neural networks with a novel probabilistic mixture model. Our hierarchical mixture neural network can represent arbitrary probability distributions, including those with coherence constraints. It is accurate, computationally efficient, and probabilistically coherent by construction.

1.4 Overview

We organize the thesis into three main parts. The first part introduces the time forecasting methodology, reviewing different models, principles, and approaches to prepare, produce and evaluate forecasts, focusing on neural forecasting methods. The second section presents four case studies showcasing our contributions to the neural forecasting literature. Finally, the third and concluding part summarizes the thesis and outlines potential avenues for future research. Below, we provide a brief overview of each part.

Part I: Background

This part introduces time series forecasting, framing it as a multivariate regression problem.

The second chapter delves deeper into the regression problem and examines predictor variables, evaluation methods, model optimization, probabilistic estimation, and model and hyperparameter selection. The third chapter overviews contemporary forecasting models, focusing on machine learning-based models and neural networks, and discusses related research. Finally, we briefly outline the technical approach employed in the case studies composing the remainder of the work.

Part II: Case Studies

Chapter 4: Interpretable Neural Forecasting

Accuracy alone is not always enough; in some cases, our ability to understand a model's forecasts is equally important. In situations like this, the need for interpretability can hinder the adoption of neural forecasting models.

In this first case study, we demonstrate how leveraging classic econometric signal decomposition techniques can improve the interpretability of neural forecast models without sacrificing their accuracy. We extend the Neural Basis Expansion analysis method (Oreshkin et al., 2020) by incorporating exogenous variables, which significantly enhances its accuracy and enables the integration of multiple sources of helpful information. The NBEATSx neural network provides an interpretable signal decomposition, allowing users to visualize the relative impact of trend and seasonal components and the interactions with exogenous factors. With NBEATSx, it becomes easier to comprehend how the model constructs its predictions.

We assess the effectiveness of NBEATSx, evaluating it on the challenging electricity price forecasting task, which has been extensively studied. Our results show that NBEATSx achieves state-of-the-art performance, improving the forecast accuracy by nearly 20% compared to the original NBEATS model and by up to 5% over other well-established statistical and machine learning methods specialized for this task.

Chapter 5: Probabilistic Hierarchical Forecasting

The hierarchical forecasting challenge arises when time series data is organized into natural groups with multiple levels of aggregation, for which we need accurate predictions that maintain *probabilistic coherence*.

In this case study, motivated by the shortcomings of existing methods, which often lack accuracy or are computationally complex, we propose a novel approach that combines the strengths of neural networks with a novel multivariate mixture model. Our composite hierarchical mixture neural network (HINT) method is accurate, computationally efficient, and probabilistically coherent by construction.

In principle, the hierarchical mixture neural network can represent arbitrary conditional probability distributions, including those with coherence constraints, in the same way, a conventional neural network can represent arbitrary functions. We demonstrate the effectiveness of the hierarchical mixture networks on three real-world hierarchical datasets; we achieve relative performance improvements of 11.8% on Australian domestic tourism data, 8.1% on the Favorita grocery store dataset, and similar results to statistical reconciliation methods on a San Francisco Bay Area highway traffic dataset.

Chapter 6: Long-Horizon Forecasting

Long-horizon forecasting remains a challenge. Recurrent predictions suffers error concatenation, while multistep predictions suffer high variance due to their over-parametrized nature.

In this case study, we tackle the high volatility and computational complexity limitations of multistep forecasting strategies introducing the neural hierarchical interpolation for time series (NHITS). It addresses these challenges by incorporating innovative hierarchical interpolation and multi-rate data sampling techniques inspired by Wavelet analysis.

By assembling predictions sequentially and emphasizing components with different frequencies and scales, NHITS significantly improves accuracy in long-horizon forecasting tasks while reducing computation time by orders of magnitude compared to existing neural forecasting approaches. We demonstrate its capabilities on six large-scale benchmark datasets from the long-horizon forecasting literature: electricity transformer temperature, exchange rate, electricity consumption, San Francisco bay area highway traffic, weather, and influenza-like illnesses, where we improve point prediction accuracy by almost 20% over the previous state-of-the-art.

1.5 Bibliographic Notes

Most of the work in this thesis has been published or is in revision in the following venues.

Chapter 4 is based on:

- Kin G. Olivares, Cristian Challu, Grzegorz Marcjasz, Rafał Weron, and Artur Dubrawski. "Neural basis expansion analysis with exogenous variables: Forecasting electricity prices with NBEATSx". In: *International Journal of Forecasting* (2022). ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2022.03.001. URL: https://www.sciencedirect.com/science/article/pii/S0169207022000413

Chapter 5 is based on:

- Kin G. Olivares, Nganba Meetei, Ruijun Ma, Rohan Reddy, Mengfei Cao, and Lee Dicker. "Probabilistic Hierarchical Forecasting with Deep Poisson Mixtures". In: *International Journal of Forecasting, submitted* Accepted Paper version available at arXiv:2110.13179 (2021). URL: https://arxiv.org/abs/2110.13179
- Kin G. Olivares, David Luo, Stefania La Vattiata, Cristian Challu, Max Mergenthaler, and Artur Dubrawski. "HINT: Hierarchical Neural Networks For Coherent Probabilistic Forecasting". In: *International Conference of Machine Learning* Workshop paper available at arXiv:2110.13179 (2023). URL: https://arxiv.org/abs/2110.13179

Chapter 6 summarizes the work on:

- Cristian Challu, Kin G. Olivares, Boris N. Oreshkin, Federico Garza, Max Mergenthaler, and Artur Dubrawski. "N-HiTS: Neural Hierarchical Interpolation for Time Series Forecasting". In: *The Association for the Advancement of Artificial Intelligence Conference 2023 (AAAI 2023)*. 2023. URL: https://arxiv.org/abs/2201.12886
- Cristian Challu, Kin G. Olivares, Gus Welter, and Artur Dubrawski. "DMIDAS: Deep Mixed Data Sampling Regression for Long Multi-Horizon Time Series Forecasting". In: *9th International Conference of Machine Learning, (ICML 2021)* Workshop on Time Series (2021). URL: https://arxiv.org/abs/2106.05860

Since Chapters 4 to 6 present previously submitted results, the work on this thesis presents a unified notation and general framework. In addition to a more concise, summarized version of the results and discussion of new research ideas.

1.6 Open-Source Contributions

Evaluating and comparing new forecasting methods with established baselines is crucial for their systematic development. For this reason, as an integral component of this thesis work, we made most of the models and baselines available in their corresponding open-source libraries.

- `NeuralForecast` (Olivares et al., 2022b): A Python library specialized in time series forecasting with deep learning models, that contains efficient datasets and data-loading utilities, evaluation functions, statistical tests implemented in `PyTorch` (Paszke et al., 2019) and `PyTorchLightning` (Falcon et al., 2019).
- `StatsForecast` (Garza et al., 2022): A python library offering widely used univariate time series forecasting models, including automatic `ARIMA`, `ETS`, `CES`, and `Theta`modeling optimized for high performance using `Numba` (Lam et al., 2015).
- `HierarchicalForecast` (Olivares et al., 2022c): A benchmark library for hierarchical forecasting that builds upon Python's fastest open-source `ETS`/`ARIMA`implementations to improve the availability, utility, and adoption of hierarchical forecast reference baselines.
- `GluonTS` (Alexandrov et al., 2020): Another Python library for deep-learning based time series modeling, mostly based on `MXNet` (Tianqi Chen et al., 2015), the package built API calls to several R baselines' implementations (Hyndman et al., 2020), as well as the `HierE2E` model (Rangapuram et al., 2021).

These are the references to the open source contributions resulting from this thesis work:

- Kin G. Olivares, Cristian Challú, Federico Garza, Max Mergenthaler Canseco, and Artur Dubrawski. *NeuralForecast: User friendly state-of-the-art neural forecasting models.* PyCon Salt Lake City, Utah, US 2022. 2022. URL: https://github.com/Nixtla/neuralforecast
- Kin G. Olivares, Federico Garza, David Luo, Cristian Challú, Max Mergenthaler, Souhaib Ben Taieb, Shanika L. Wickramasuriya, and Artur Dubrawski. "HierarchicalForecast: A Reference Framework for Hierarchical Forecasting in Python". In: *Work in progress paper, submitted to Journal of Machine Learning Research.* abs/2207.03517 (2022). URL: https://arxiv.org/abs/2207.03517
- Federico Garza, Max Mergenthaler Canseco, Cristian Challú, and Kin G. Olivares. *StatsForecast: Lightning Fast Forecasting with Statistical and Econometric Models.* PyCon Salt Lake City, Utah, US 2022. 2022. URL: https://github.com/Nixtla/statsforecast

Part I

Forecasting Principles

I am tomorrow, or some future day, what I establish today. I am today what I established yesterday or some previous day.

James Joyce

Here we introduce the forecasting task, its principles, and its practice with a look into the future. In Chapter 2, we provide an introduction to the regression problem, forecast evaluation methods, as well as common model estimation and optimization techniques. In Chapter 3, we briefly overview forecast modeling approaches, including statistical, econometric, and machine learning techniques, that served as a reference for the thesis work.

Chapter 2

Background

2.1 What is Time Series Forecasting?

Forecasting has always been at the forefront of decision making and planning. Its theory and models rest on the premise that by analyzing historical data, patterns can be identified and utilized to make accurate predictions about the future values of a time series. Forecasting aims to predict the future as accurately as possible, conditional on all the available information.

A time series is a sequence of chronological observations of a random variable. When the observations are uniformly spaced, they comprise a regular time series; when the space between the observations varies, they comprise an irregular time series. In this work, we will focus on the first kind of series.

In this thesis we denote the *target time series* as $\mathbf{y}$, it can be univariate or multi variate. We denote the *forecast creation date* with the time index t, in which the prediction is created. For any forecast creation date t, within the forecast horizon of size H, we denote the relative *prediction time step* with $\tau \in [t+1, \dots, t+H]$.

We denote *point predictions* that estimate the central location of the target time series' future through a mean or median value as $\hat{\mathbf{y}}_\tau$, with the following vectorized notation:

$$\mathbf{y}_{[t+1:t+H]} = [\mathbf{y}_{t+1}, \dots, \mathbf{y}_{t+H}] \quad \text{and} \quad \hat{\mathbf{y}}_{[t+1:t+H]} = [\hat{\mathbf{y}}_{t+1}, \dots, \hat{\mathbf{y}}_{t+H}] \tag{2.1}$$

In case of *probabilistic predictions* that conveys the uncertainty around the central forecast, we denote the *forecasting probability* as $\hat{\mathbb{P}}$. In time series forecasting, uncertainty plays a crucial role in acknowledging and accounting for the inherent unpredictability of future events, which can impact the accuracy and reliability of predictions. By incorporating uncertainty, forecasters can provide more accurate and realistic forecasts, accounting for a range of possible outcomes.

$$\hat{\mathbb{P}}(\mathbf{y}_{[t+1:t+H]}) \tag{2.2}$$

2.2 Predictor Variables

The forecasting methods depend heavily on the availability and quality of data. We provide a classification of the most common predictor variables in terms of their relationship to the target variable and its time dependence.

2.2.1 Autoregressive Features

Basic time series foresting models use only information of the target variable and omit the attempt to learn predictive relationships with other variables. Despite their simplicity, the univariate time series forecasting methods include well-proven methods like Naive, Seasonal Naive, ETS, ARIMA, discussed in Section 3.1. These methods should always be considered as baselines to evaluate the predictive accuracy of other more complex alternatives. We refer to the past values of the target series as *autoregressive features* and denote them through the thesis with the following notation:

$$\mathbf{y}_{[t-L:t]} = [\mathbf{y}_{t-L}, \mathbf{y}_{t-L+1}, \dots, \mathbf{y}_{t-1}, \mathbf{y}_{t}] \quad \text{and} \quad \mathbf{y}_{[:t]} = [\dots, \mathbf{y}_{t-L}, , \mathbf{y}_{t-L+1}, \dots, \mathbf{y}_{t-1}, \mathbf{y}_{t}] \tag{2.3}$$

2.2.2 Exogenous Variables

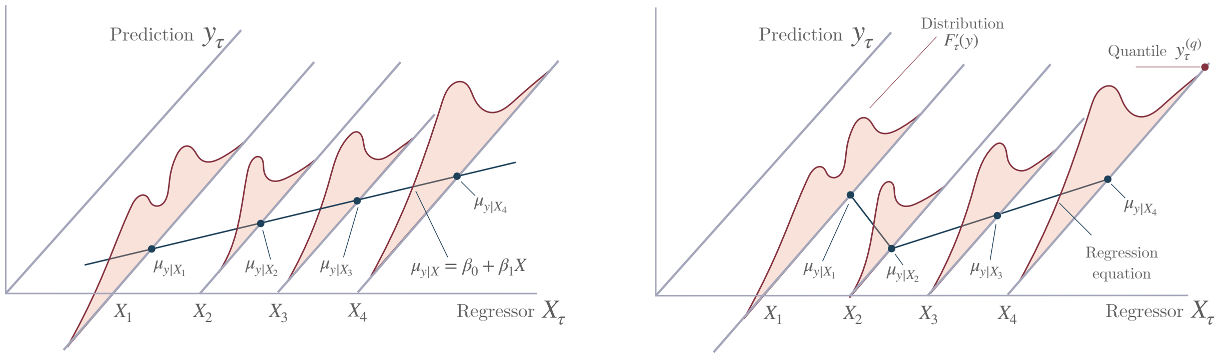
We refer to the features used to create the predictions beyond the autoregressive as exogenous variables. We distinguish three types of exogenous variables depending on whether they reflect static or time-dependent aspects of the modeled data and their availability at the time of the predictions t, into static, historical, and future exogenous variables.

Static. The *static exogenous* variables $\mathbf{x}^{(s)}$ carry time-invariant information. When the models share parameters across multiple time series, these variables allow sharing information within groups of time series with similar static variable levels. Examples of static variables include designators such as identifiers of regions, groups of products, etc., that mark agglomerates of series that demonstrate similar behaviors.

Historic. The *historic exogenous* variables $\mathbf{x}^{(h)}_{[:t]}$ provide the models with information at the moment of the forecast creation that are determined independently, and they are not systematically affected by the target variables yet affect it. Examples of these variables can be treatments in healthcare-related data prices of goods and services in the case of competitive markets. Another example is weather data for the prediction of agricultural yield, as weather remains unaffected by local small production. Or advertisement spending, where we normally assume control, and treat it as independent from demand. Endogenous variables pose different challenges as they determine multivariate forecasting problems.

Future. The *future exogenous* variables $\mathbf{x}^{(f)}_{[:t+H]}$ provide the models of available information about the future, planned events, special events, or even predictions of other covariates. An example of these features can be seasonal covariates linked to the natural frequencies in the data encoded in calendar variables to identify hours, days, months, or holidays, among others. Another example can be the distance to events like holidays or distance to promotions in retail and e-commerce.

2.3 Forecasting Task



(a) Linear **(b)** General

Figure 2.1: Multivariate Regression

The time-series forecasting task that we tackle in this thesis can be formally represented with a variant of the following high-dimensional multivariate regression problem:

$$\mathbb{P}(\mathbf{y}_{[t+1:t+H]} \mid \mathbf{X}_{[:t+H]}) \quad \text{with} \quad \mathbf{X}_{[:t+H]} = \{\mathbf{y}_{[:t]},\ \mathbf{x}^{(h)}_{[:t]},\ \mathbf{x}^{(f)}_{[:t+H]},\ \mathbf{x}^{(s)}\} \tag{2.4}$$

where, for description simplicity, $\mathbf{X}_{[:t+H]}$ denotes the collection of predictor variables that contain, autorregressives, historic, future and static exogenous variables. Figure 2.1 shows the univariate regression case with a single future exogenous variable.

Probabilistic Forecasting. Making predictions for the future involves varying degrees of imperfect or unknown information that translates into uncertainty (Dawid, 1984). For this reason, probabilistic forecasting has emerged as a natural answer to quantify the uncertainty of the future of the target variable, conditioning on the available information of its predictors. The probabilistic forecasting task is to produce at any time t a *predictive probability distribution* for the next observations of the target variable $\mathbf{y}_{[t+1:t+H]}$. A consequence of the estimation of the joint probability distribution throughout the horizon are the marginal distributions $\hat{F}_\tau$ for each prediction time step $\tau \in [t+1:t+H]$, Figure 2.1b depicts them. Equation (2.5) defines the predictive marginal distribution and its quantiles, that compose prediction intervals.

$$\hat{F}_\tau(y) = \hat{\mathbb{P}}\left(y_\tau \leq y \mid \mathbf{X}_{[:t+H]}\right) \qquad \hat{y}^{(q)}_\tau = \hat{F}^{-1}_\tau(q) \tag{2.5}$$

Point Forecasting. For a long time, statisticians and forecasters relied on Gaussian assumptions and treated forecasts as an expression of the information of its parameters, particularly its location. The point forecasting task is to produce an estimation of the future location of the target variable y_τ, and the model's output is often the conditional mean from Equation (2.6) or a robust median as a particular case of Equation (2.5).

$$\hat{\mu}_{y_\tau|\mathbf{X}_\tau} = \mathbb{E}\left[\, y_\tau \mid \mathbf{X}_{[:t+H]} \,\right] \tag{2.6}$$

2.4 Evaluation

2.4.1 Point Forecast Errors

The most important magnitude for point forecast is the forecast error, which is the difference between the observed value y_τ and the prediction $\hat{y}_\tau$, at time τ:

$$e_\tau = y_\tau - \hat{y}_\tau \qquad \text{with} \qquad \tau \in [t+1:t+H] \tag{2.7}$$

The forecasting community tends to differentiate between that forecast errors from regression residual, in the sense that we measure forecast errors in the validation and test sets. In contrast, we measure regression residuals in the train set that we define in Section 2.4.4. Finally, forecast accuracy summarizes the errors in different metrics that we will explain below. We follow closely Hyndman and Athanasopoulos, 2018b taxonomy, with the addition of the probabilistic errors.

1. **Scale-dependent errors**. This type of measurement is on the same scale as the data, for which it is desirable that the data is normalized or the scales of the time-series that compose it are comparable. The most common metrics are Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE) that is more robust to outliers, than its unrooted counterpart.

$$\mathrm{MAE} = \frac{1}{H}\sum_{\tau=t+1}^{t+H} |y_\tau - \hat{y}_\tau| \tag{2.8}$$

$$\mathrm{MSE} = \frac{1}{H}\sum_{\tau=t+1}^{t+H} (y_\tau - \hat{y}_\tau)^2 \qquad \mathrm{RMSE} = \sqrt{\frac{1}{H}\sum_{\tau=t+1}^{t+H} (y_\tau - \hat{y}_\tau)^2} \tag{2.9}$$

2. **Percentage errors**. Percentage errors have the advantage of being unit-free, making them suitable for comparison across datasets or time series of different scales. Common percentage metrics are Mean Absolute Percentage Error (MAPE), symmetric Mean Absolute Percentage Error (sMAPE) (Meade and Armstrong, 1986). Hyndman and Koehler 2006 recommend sMAPE to avoid MAPE's degradation around y_τ zero.

$$\mathrm{MAPE} = \frac{1}{H}\sum_{\tau=t+1}^{t+H} \frac{|y_\tau - \hat{y}_\tau|}{|\hat{y}_\tau|} \qquad \mathrm{sMAPE} = \frac{200}{H}\sum_{\tau=t+1}^{t+H} \frac{|y_\tau - \hat{y}_\tau|}{|y_\tau| + |\hat{y}_\tau|} \tag{2.10}$$

3. **Relative/Scaled errors**. Relative accuracy measures offer a way to compare the prediction errors to baseline models, examples are relative Mean Absolute Error (relMAE) (Hyndman and Koehler, 2006; Lago et al., 2021a) and relative Mean Squared Error (relMSE) (Olivares et al., 2021). As in percentage errors, the measure is unit-free. When the error is greater than one, the predictions are worse than the baseline's predictions

$$\mathrm{relMSE} = \frac{\sum_{\tau=t+1}^{t+H} (y_\tau - \hat{y}_\tau)^2}{\sum_{\tau=t+1}^{t+H} (y_\tau - \hat{y}^{base}_\tau)^2} \qquad \mathrm{relMAE} = \frac{\sum_{\tau=t+1}^{t+H} |y_\tau - \hat{y}_\tau|}{\sum_{\tau=t+1}^{t+H} |y_\tau - \hat{y}^{base}_\tau|} \tag{2.11}$$

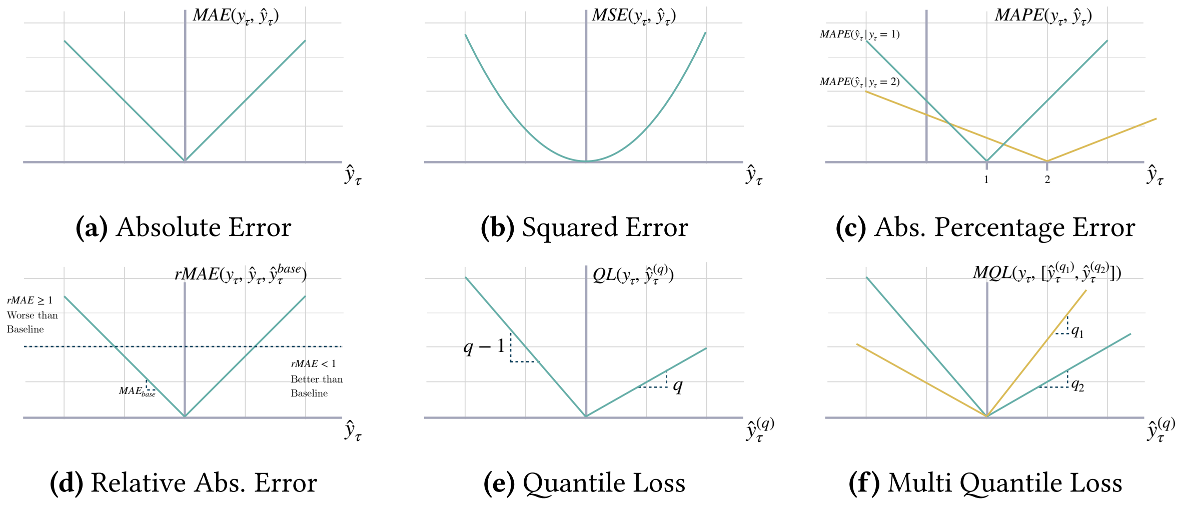


Figure 2.2: Evaluation Metrics and Train Losses

2.4.2 Probabilistic Forecast Errors

Here we describe metrics that allow to empirically asses the accuracy and compare *probabilistic forecasts* introduced in Section 2.3.

1. **Quantile Loss**. Consider the estimated cumulative distribution function $\hat{\mathbf{F}}_\tau$ and its associated quantiles $\hat{y}_\tau^{(q)} = \hat{\mathbf{F}}_\tau^{-1}(\,q\,)$, for an observation y_τ, the Quantile Loss (QL) (Matheson and Winkler, 1976), depicted in Figure 2.2e, is defined as:

$$\begin{aligned}\mathrm{QL}(y_\tau, \hat{y}_\tau^{(q)}) &= 2\Big((1-q)\,(\hat{y}_\tau^{(q)} - y_\tau)_+ + q\,(y_\tau - \hat{y}_\tau^{(q)})_+\Big)\\ &= 2\left(\mathbb{1}\{y_\tau \leq \hat{\mathbf{F}}_\tau^{-1}(\,q\,)\} - q\right)\left(\hat{\mathbf{F}}_\tau^{-1}(\,q\,) - y_\tau\right)\end{aligned} \tag{2.12}$$

2. **Multi Quantile Loss**. The Multi Quantile Loss (MQL) (Wen et al., 2017), depicted in Figure 2.2f, measures simultaneously the errors for various estimated quantiles.

$$\mathrm{MQL}(y_\tau, [y_\tau^{(q_1)}, \dots, y_\tau^{(q_Q)}]) = \frac{1}{Q}\sum_{q_i} \mathrm{QL}(y_\tau, y_\tau^{(q_i)}) \tag{2.13}$$

3. **Continuous Ranked Probability Score**. Additionally when the objective a the full predictive distribution a common evaluation metric is the Continuous Ranked Probability Score (CRPS) (Matheson and Winkler, 1976). The CRPS measures the accuracy of whole predictive distributions and has desirable theoretical properties as a metric (Gneiting and Ranjan, 2011). Following notation from Laio and Tamea 2007, the CRPS is defined as:

$$\mathrm{CRPS}(y_\tau, \hat{\mathbf{F}}_\tau) = \int_0^1 \mathrm{QL}(y_\tau, \hat{y}_\tau^{(q)})_q \, dq \tag{2.14}$$

The evaluation of the CRPS uses numerical integration technique, that discretizes the quantiles and treats the integral with a left Riemann approximation, averaging over uniformly distanced quantiles, as the asymptotic behavior of the MQL.

2.4.3 Prediction's Improvements Statistical Tests

To assess which forecasting model provides better predictions, we rely on the Giacomini-White test (GW) (Giacomini and White, 2006) of the multi-step conditional predictive ability, which can be interpreted as a generalization of the Diebold-Mariano test (DM) (Diebold and Mariano, 2002), widely used in the forecasting literature. Compared with the DM or other unconditional tests, the GW test is valid under general assumptions such as heterogeneity rather than stationarity of data. The GW test examines the null hypothesis of equal accuracy specified in Equation (2.15), measured by the MAE or $L1$ norm of the forecast errors of a pair of models A and B, conditioned on the available information to that moment in time $\mathcal{F}_t$.

$$H_0 : \mathbb{E}\left[||\mathbf{y}_\tau - \hat{\mathbf{y}}_\tau^A||_1 - ||\mathbf{y}_\tau - \hat{\mathbf{y}}_t^B||_\tau \mid \mathcal{F}_t\right] \equiv \mathbb{E}\left[\Delta_t^{A,B} \mid \mathcal{F}_t\right] = 0 \tag{2.15}$$

The available information $\mathcal{F}_t$ is usually replaced with a constant and lags of the error difference $\Delta_t^{A,B}$ and the test is performed using a linear regression with a Wald-like test. When the conditional information is only the constant variable, one recovers the original DM test.

2.4.4 Train, Validation and Test splits

The magnitude of the train errors rarely provide a good assessment of the future generalization ability of the model; this becomes evident with modern, flexible models prone to overfit. A reliable estimator for the accuracy of a model's predictions requires that the model obtains and produces genuine forecast signals during model optimization and evaluation, avoiding possible future information leakage into the model's inputs during train and inference. A temporal train-evaluation split procedure allows us to estimate the model's generalization performance on future data unseen by the model. We use the *train set* to optimize the model parameters, and the *validation and test sets* to evaluate the accuracy of the model's predictions. The difference between the validation and test sets is that the validation is used during hyperparameter optimization, while we reserve the *test set* for the final measurements. Figure 2.3 shows an example.

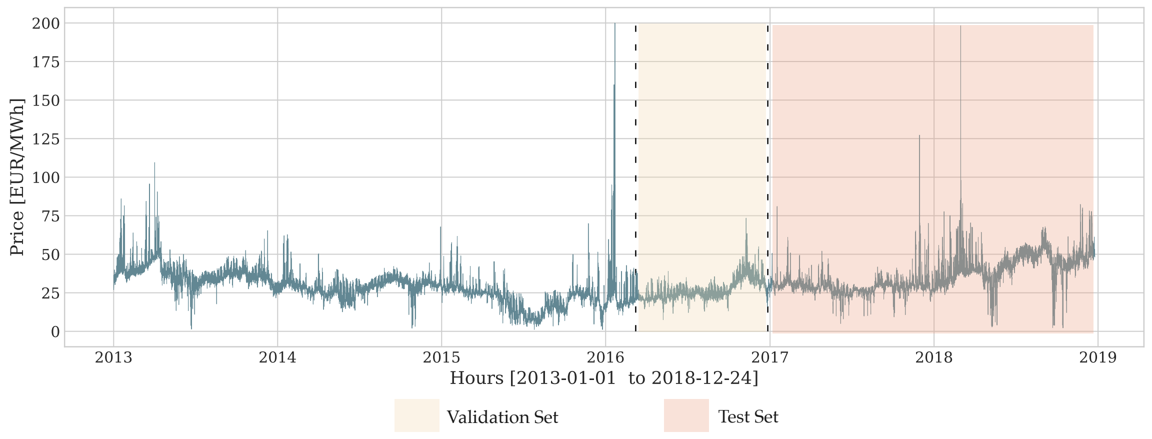
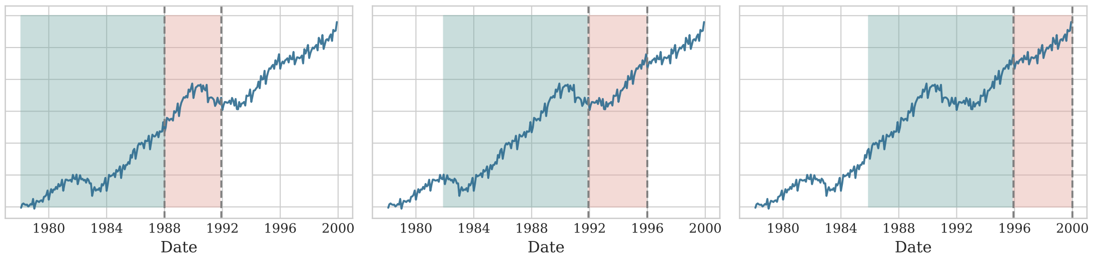
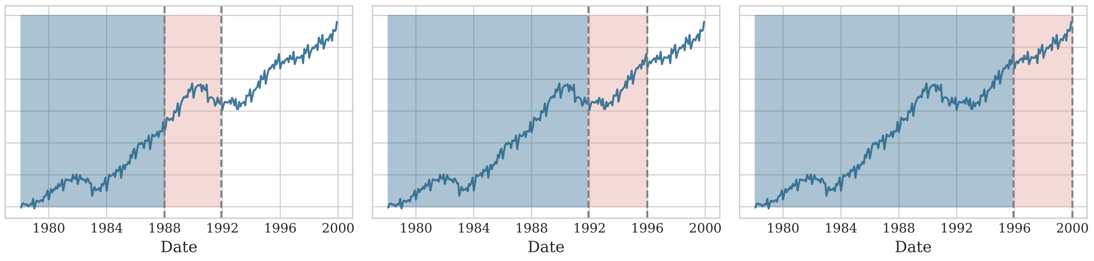


Figure 2.3: Time-series Data Splits

2.4.5 Temporal Cross-Validation



(a) Rolling Windows



(b) Chaining Windows

Figure 2.4: Temporal cross-validation assesses a model's forecast accuracy over time. It sequentially defines a sliding or chaining windows (green and blue) and test windows (orange).

Usually, the test set is longer than the forecast horizon H, which allows for an improved time-series cross-validation version over the train and evaluation set split. Cross-validation improves the forecast accuracy estimation by reducing its variability through using different data partitions (Arlot and Celisse, 2010). A particular version of the technique known as time-series cross-validation, that avoids future information leakage, is commonly used (Hyndman and Athanasopoulos, 2018b). This technique creates multiple training windows consisting only of observations prior to the test windows. The final forecast evaluation is the average of the errors on the chained or rolled test windows.

Figure 2.4 depicts time-series cross-validation. We mark the train observations in green and blue, while the test observations are orange. In this example, the final forecast evaluation average would be done across the t_1, t_2, t_3 indexes. The time-series cross-validation can be a rolling or chaining window strategy depending on the model's inputs. Figure 2.4a depicts rolling windows of size L that corresponds to the lag inputs $\mathbf{y}_{[t-L:t]}$, while Figure 2.4b represents models with infinite theoretical memory capabilities, with all the past inputs $\mathbf{y}_{[:t]}$ available until time t.

Model Recalibration. In practical settings, it is advisable to retrain the model after updating the rolling/chaining windows to incorporate all the available data before the predictions. In the time-series forecasting tasks, this helps with rapidly shifting distributions. The forecast accuracy gains come with the downside of this strategy is the additional computational costs of the model optimization. We can avoid the recalibration process for more extended periods in the presence of slowly shifting distributions.

2.5 Model Estimation

A model approximates the relationships between the predictor variables and the target variables through a set of assumptions. Most of times it is defined by a family of functions $\mathcal{F} = \{f\}$ parametrized by a vector $\theta \in \Theta$, and the most common way to estimate its values is through an *objective loss function* $\mathcal{L} : \mathcal{Y} \times \mathcal{X} \times \Theta \rightarrow \mathbb{R}^{+}$, that depends on training data $\mathcal{D} = \{(\mathbf{y}_{[t+1:t+H]}, \mathbf{X}_{:t+H})\}$. We further simplify the notation of the subsection and denote the target variable as $\mathbf{y} = \mathbf{y}_{[t+1:t+H]}$ and the predictors as $\mathbf{X} = \mathbf{X}_{:t+H}$. Finding good performing parameters in the training data is also known as fitting, or training a model

2.5.1 Empirical Risk Minimization

In general a learning task can be translated into a function estimation problem using the classic Empirical Risk Minimization (ERM) framework (Vapnik, 1999), where the objective is to minimize the expected loss function at the dataset level:

$$\hat{\theta} := \underset{\boldsymbol{\theta} \in \Theta}{\arg\min}\, \mathbb{E}\left[\, \mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}(\mathbf{X}|\, \boldsymbol{\theta}))\,\right] \quad \text{and} \quad f^{*} := \underset{f \in \mathcal{F}}{\arg\min}\, \mathbb{E}\left[\, \mathcal{L}(\mathbf{y}, f(\mathbf{X}))\,\right] \tag{2.16}$$

In the particular case of the *point forecasting* task or *probabilistic forecasting* with prediction intervals, it is immediate to translate the evaluation metrics described in Section 2.4.1 and Section 2.4.2 into training losses. Here we inspect two well known regression settings.

1. **Ordinary Least Squares.** When the objective function is the mean of the squared errors or MSE, we obtain a classical regression setting that is characterized with very appealing theoretical guarantees. For instance, it can be shown under stationarity conditions that the population's risk minimizer is the conditional expectation in Equation (2.17).

$$f^{*}(\mathbf{X}) = \underset{f \in \mathcal{F}}{\arg\min}\, \mathbb{E}\left[\, \mathrm{MSE}\left(\mathbf{y}, f(\mathbf{X})\right)\right] = \mathbb{E}\left[\, \mathbf{y} \,|\, \mathbf{X}\,\right] \tag{2.17}$$

2. **Quantile Regression.** When the objective function is the mean of the quantile loss (QL), we obtain a quantile regression setting (Koenker and Bassett, 1978). These estimators offer a non-parametric alternative for the estimation of the prediction intervals. They have essential connections to robust statistics, as the objective function offers a very desirable resistance to outliers while being as efficient as MSE estimation. Additionally, the estimators also have very appealing theoretical guarantees. Under stationarity conditions, we can show that the population's risk minimizer is the conditional quantiles in Equation (2.18).

$$f^{*}(\mathbf{X}) = \underset{f \in \mathcal{F}}{\arg\min}\, \mathbb{E}\left[\, \mathrm{QL}_{q}\left(\mathbf{y}, f(\mathbf{X})\right)\right] = \mathbf{F}_{\mathbf{y}}^{-1}\left(\, q\,\right) \tag{2.18}$$

A lot of research has been done recently on model estimation, in particular when combined with regularization side objectives that help to restrict the flexibility of the family of functions $\mathcal{F}$ into a better behaved spaced based. Examples of this include RIDGE and LASSO regressions (Hastie et al., 2009).

2.5.2 Maximum Likelihood Estimation

Advancing the construction forecasting models, one can assume the target variable $\mathbf{y} = \mathbf{y}_{[t+1:t+H]}$ coming from a particular probability distribution for Equation (2.4). This approach has many advantages. The output of the model is a complete *predictive distribution* rather than only location estimates or prediction intervals. The selection of the probability distribution can encode domain expertise that can help the accuracy of the models, for example, when constraining the event space of the probability to guarantee consistency of the predictions.

The likelihood intends to measure the probability of the data $\mathcal{D}$ of being originated by the model; a large likelihood is linked with good models, while small likelihoods are associated with poor models. To estimate the parameters of these probabilistic forecasting models, one can minimize the negative log-likelihood of the data under the maximum likelihood estimation Maximum Likelihood Estimation (MLE) (Murphy, 2012) optimization strategy:

$$\hat{\boldsymbol{\theta}} := \underset{\boldsymbol{\theta}\in\Theta}{\arg\min}\, \mathbb{E}\left[\,\mathcal{L}(\mathbf{y}, \mathbf{X}, \boldsymbol{\theta})\,\right] = \mathbb{E}\left[\,-\log\left(\,\mathbb{P}(\,\mathbf{y} \mid \mathbf{X}, \boldsymbol{\theta})\,\right)\right] \tag{2.19}$$

For a very long time the forecasting community used the least squares errors estimation method as a default. This method under restrictive assumptions, namely that the regression is linear in its parameters, stationarity and no multi-collinearity of the predictors, homoscedasticity, normally distributed errors centered around zero, matches the MLE optimization strategy with Gaussian distribution. Since then a lot of progress has been done beyond Gaussian assumptions.

2.5.3 Maximum a Posteriori Estimation

A closely related alternative to MLE is the Maximum a Posteriori Estimation (MAP) strategy that employs an augmented optimization objective that can often be interpreted as regularization. The main intuition behind it is the additional assumption of a probability distribution for the parameters $\mathbb{P}(\boldsymbol{\theta})$ that constraints the likelihood optimization. MAP estimation method is defined by Equation (2.20):

$$\hat{\boldsymbol{\theta}} := \underset{\boldsymbol{\theta}\in\Theta}{\arg\min}\, \mathbb{E}\left[\,\mathcal{L}(\mathbf{y}, \boldsymbol{\theta})\,\right] = \mathbb{E}\left[\,-\log\left(\,\mathbb{P}(\,\mathbf{y} \mid \mathbf{X}, \boldsymbol{\theta})\,\right) - \log\left(\,\mathbb{P}(\boldsymbol{\theta})\,\right),\right] \tag{2.20}$$

Regarding the fully-Bayesian approach we think there are still plenty of research opportunities to improve its scalability, since many practical scenarios the forecasting needs are in the hundreds or thousands of time series, we do not consider it in this work.

The three mentioned methods are have been used with success in forecasting applications. They have their benefits and disadvantages regarding their predictive accuracy, computational complexity, and theoretical support. In this work Chapter 4 and Chapter 6 tackle the *point forecasting task* and use an empirical risk minimization approach, while Chapter 5 tackles a *probabilistic forecasting task* and extends the maximum likelihood estimation towards composite likelihood (Lindsay, 1988; Varin et al., 2011) with to improve its computational feasibility.

2.5.4 Parameter Optimization

The process of finding a maximizing or minimizing parameter for the loss functions introduced in Section 2.5 is known as optimization. When the *objective loss function* is characterized with advantageous smoothness properties like differentiability or subdifferentiability, one can rely on the long tradition of gradient-based optimization algorithms (Cauchy et al., 1847). Through time Stochastic Gradient Descent (Robbins and Monro, 1951) has become one of the most important optimization methods in Machine Learning because of its efficiency in respect to the size of the datasets and model's parameters is uncontested. We describe SGD in Algorithm 2.1.

Algorithm 2.1 Stochastic Gradient Descent.

```
1: $\boldsymbol{\theta}_1 \in \Theta$, learning rate $\alpha$.
2: repeat
3:     Sample a training batch $(\mathbf{y}_i, \mathbf{X}_i) \subset \mathcal{D}$.
4:     Calculate the gradient $\mathbf{g}_t = \nabla \mathcal{L}(\mathbf{y}_i, \mathbf{X}_i, \hat{\boldsymbol{\theta}}_t)$.
5:     Update parameters $\hat{\boldsymbol{\theta}}_{t+1} = \hat{\boldsymbol{\theta}}_t - \alpha \mathbf{g}_t$
6: until convergence
```

Improvements on SGD include momentum-based methods built by the contributions of Nesterov's accelerated gradient (Nesterov, 1983), that under better smoothness conditions of the objective function tends to dampen the oscillations of SGD and speed up its iterations, thus achieving a faster convergence. A classic intuition on momentum is that it adds inertia to the gradients acting as a trajectory smoother that traverses local minima and small increases of the loss function.

One of the most popular gradient optimization variants is Adaptive Moment Estimation SGD (ADAM) (Kingma and Ba, 2014), described in Algorithm 2.2. In addition to the gradient's momentum ADAM includes second-order moments to rescale the gradients and implicitly adapt the learning rate for different parameters. ADAM can be thought of as performing a diagonal approximation to the Hessian that accounts for the local curvature of the objective in classic second-order optimization methods. The method is fairly robust to the learning rate selection and has very fast convergence. In this thesis we mostly rely on this optimization method.

Algorithm 2.2 Adaptive Moment Estimation SGD.

```
1: $\boldsymbol{\theta}_1 \in \Theta$, learning rate $\alpha$, $\beta_1, \beta_2$ decay rates for moments.
2: repeat
3:     Sample a training batch $(\mathbf{y}_i, \mathbf{X}_i) \subset \mathcal{D}$.
4:     Calculate the gradient $\mathbf{g}_t = \nabla \mathcal{L}(\mathbf{y}_i, \mathbf{X}_i, \hat{\boldsymbol{\theta}}_t)$.
5:     Estimate first moment $\mathbf{m}_t = \beta_1 \mathbf{m}_{t-1} + (1 - \beta_1)\mathbf{g}_t$
6:     Estimate second moment diagonal $\mathbf{D}_t = \beta_2 \mathbf{D}_{t-1} + (1 - \beta_2)\mathbb{I} \odot \mathbf{g}_t^2$
7:     Correct the moments bias $\mathbf{m}_t = 1/(1 - \beta_1^t)\mathbf{m}_t$ and $\mathbf{D}_t = 1/(1 - \beta_2^t)\mathbf{D}_t$
8:     Update parameters $\hat{\boldsymbol{\theta}}_{t+1} = \hat{\boldsymbol{\theta}}_t - \alpha\, \hat{\mathbf{D}}_t^{-1/2} \mathbf{m}_t$
9: until convergence
```

2.6 Model Selection

2.6.1 Model Identification

The process of model identification, defined for many years the forecasting practice, as most statisticians and forecasters pursued the discovery of a "true model" governing the data generation process of the time series. The process of model selection was that of the definition of the assumptions of relatively simple models that governed the trend, seasonal and autoregressive true relationships on the target variable. Rather than a more pragmatic approach to minimize the predictive errors in a held-out dataset (Bates and Granger, 1969).

Automatic forecasting methods inherited the identification tradition, and explore classical families like Error, Trend, Seasonality Exponential Smoothing State Space (ETS) (Holt, 1957; Hyndman et al., 2008) and Autorregresive Integrated Moving Average (ARIMA) in a statistically principled way to ensure stationarity of the modeled processes, seasonalities and autoregressive features (Box et al., 2015a; Hyndman and Khandakar, 2008). These methods offer a solution to forecasting tasks of large numbers of univariate time series and are obligated baselines.

2.6.2 Hyperparameter Optimization

Machine Learning forecasting methods are defined by many hyperparameters that control their behavior, with effects ranging from their speed and memory requirements to their predictive performance. For a long time, hyperparameter tuning was done manually. This approach is time-consuming, and automated, efficient hyperparameter optimization methods have been introduced, proving to be more efficient than manual tuning, grid search, and random search. The comprehension of the impacts of the hyperparameters is still a precious skill, as it can help guide the design of informed hyperparameter spaces that are faster to explore automatically.

A common approach for hyperparameter optimization is a technique called sequential model-based optimization that intelligently samples hyperparameter configurations deemed promising based on the validation evaluation of other hyperparameter configurations. Gaussian Processes or adaptive Parzen windows approximate the expected improvement (EI) for new configurations. Algorithm 2.3 describes the Sequential Model-Based Hyperparameter Optimization (HYPEROPT) (Bergstra et al., 2011). All novel methods proposed in this thesis rely on HYPEROPT to select configurations with the added benefit of guaranteeing the research's replicability.

Algorithm 2.3 Sequential Model-based Hyperparameter Optimization.

```
1: Initialize $\mathcal{H} = \emptyset$ configurations' evaluations, and approximation model $M_0$.
2: for each evaluation step $t \in [:T]$ do
3:     Obtain promising hyperparameters $H^* = \arg\max_H \mathrm{EI}(H, M_{t-1})$.
4:     Evaluate loss function $\mathcal{L}(H^*) = \mathcal{L}(\mathbf{y}, \mathbf{X}, \boldsymbol{\theta}(H^*))$.
5:     Update the configurations' evaluations history $\mathcal{H} = \mathcal{H} \cup \{\, (H^*, \mathcal{L}(H^*)) \,\}$.
6:     Fit a new model $M_t$ on $\mathcal{H}$
7: end for
```

Chapter 3

Overview of Modeling Approaches

In this chapter we overview the most adopted modeling approaches, using the two primary model families' classification presented by Petropoulos et al., 2022 survey as a starting point.

Statistical and econometric forecasting models rely on theory-inspired assumptions about variables' relationships. In contrast, machine learning models assemble predictions primarily based on data without being explicitly programmed, making them exceptionally flexible and with minimal assumptions. The categorization of forecasting methods into statistical and machine learning is based on whether they prescribe the data-generating process. Methods such as Neural Networks, Decision Trees, Support Vector Machines, and Gaussian Processes that build on unstructured, non-linear regression algorithms are typically categorized as machine learning models (Makridakis et al., 2018a; Januschowski et al., 2020). We do not subscribe entirely to the methods' dichotomy, and as we showcase throughout this thesis work, there are plenty of possible, enriching interactions between the two model families, although this taxonomy is a good starting point for an overview of modeling approaches.

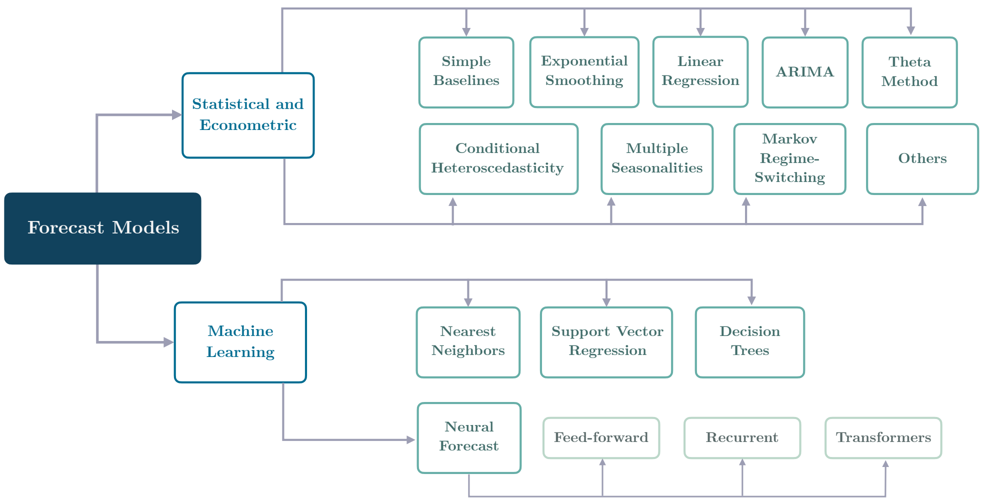


Figure 3.1: A taxonomy of time series forecasting models. We review them from Section 3.1-Section 3.3 with a special emphasis on neural forecasting models that are closely related to this thesis contributions.

3.1 Statistical and Econometric Models

3.1.1 Simple Baselines

Simple methods can be surprisingly accurate in many situations. These methods always constitute useful baselines against which we can compare the performance of more sophisticated methods. Examples of simple baselines include Naive (Naive), a random walk model that predicts using the last available observation, Seasonal Naive (SNaive) similar to Naive but using the last seasonal observation, Moving Average (MA) an average of the last L available observations, Historic Average (HA) a simple average of all past observations.

$$
\begin{aligned}
\hat{y}^{\text{Naive}}_{t+h} \mid_{t} &= y_t & \qquad \hat{y}^{\text{SNaive}}_{t+h} \mid_{t} &= y_{t+h-s\lfloor 1+\frac{h-1}{s} \rfloor} \\
\hat{y}^{\text{MA}}_{t+h} \mid_{t} &= \frac{1}{L}\sum_{l=1}^{L} y_{t-l} & \qquad \hat{y}^{\text{HA}}_{t+h} \mid_{t} &= \frac{1}{T}\sum_{l=1}^{T} y_{t-l}
\end{aligned}
\tag{3.1}
$$

For additional information on the topic we refer to the simple forecasting methods chapter from Hyndman and Athanasopoulos, 2018b book.

3.1.2 Exponential Smoothing

The idea behind Exponential Smoothing (ES) is simple and attractive. The technique creates the forecasts as weighted averages of observations, with the more recent having greater weight than those in the distant past. The exponential smoothing model family is a useful and dependable model for many practical applications. Its history dates back to 1944 with the work of Brown, 1959 and was naturally extended to smooth different signal's components like the level, trend, and seasonal patterns by Holt, 1957 and Winters, 1960.

The simplest exponential smoothing has the following recursive representation:

$$\hat{y}_{t+1} \mid_{t} = \alpha y_t + (1-\alpha)\hat{y}_{t-1} \tag{3.2}$$

where the exponential smoothing parameter $\alpha \in (0, 1)$. In the past the estimation of the ES parameters was done through discounted least squares, modern implementations minimize the squared errors of one step forward predictions (Nelder and Mead, 1965).

The method's theoretical understanding and its use cases grew, and connections between the ES family and ARIMA were stablished (Gardner Jr, 1985). Similarly, the ES family was extended with a statistical framework capable of producing forecast distributions through the Error, Trend, Seasonality Exponential Smoothing State Space (ETS) (Hyndman et al., 2008). For a taxonomy and systematic reviews of the model family, we refer to the work of Pegels, 1969; Gardner Jr, 1985; Taylor, 2003; Hyndman et al., 2008. A more recent extension of the model family includes the Complex Exponential Smoothing (CES), moves away from the classic level, trend, and seasonality decomposition, through the usage of complex-valued functions (Svetunkov, 2016).

3.1.3 Linear Regression

Classic regression is one of the most popular statistical forecasting techniques. It aims to estimate the relationship between the target time series variable y_t and its predictor variables $x_{i,t}$. In its basic form, the model is optimized by minimizing the sum of squared errors, and it can be interpreted with the following conditional Gaussian linear model:

$$y_{t+1} \mid {}_t = \theta_0 + \theta_1 x_{1,t} + \cdots + \theta_k x_{k,t} + \epsilon_t \tag{3.3}$$

$$\hat{y}_{t+1} \mid {}_t = \hat{\theta}_0 + \hat{\theta}_1 x_{1,t} + \cdots + \hat{\theta}_k x_{k,t} \tag{3.4}$$

where ϵ_t is the residual error at time t, θ_i are the coefficients that measure the effect of the predictor $x_{i,t}$ on the target variable. As mentioned in Section 2.2.2, the predictor variables can be autoregressive features, promotions, calendar dummies marking special days or holidays, and polynomial and harmonic functions to capture trends and seasonalities. This model is optimized minimizing Mean Squared Error (MSE) that matches the Maximum Likelihood Estimation (MLE) through the Gaussian distribution assumption on the residuals. When the regression models use autoregressive features, it can be interpreted as a member of the Autorregresive Models (AR) family. The community generally distinguishes the Machine Learning approach from classic regression by the relaxation of the linearity assumption of Equation (3.3).

3.1.4 Autoregressive Integrated Moving Average

Autorregresive Integrated Moving Average (ARIMA) along with Exponential Smoothing (ES) are workhorses of the forecasting practice. While exponential smoothing models rely on characterizing the trend and seasonality of the data, ARIMA models focus on characterizing the autocorrelations present in the data.

An autoregressive model predicts the target variable with a linear combination of its past. The name ”autoregression” denotes that the model is a form of regression where the variable is regressed against itself. An autoregressive model of order p, denoted as AR(p), can be written as:

$$y_t = c + \theta_1 y_{t-1} + \theta_2 y_{t-2} + \cdots + \theta_p y_{t-p} + \epsilon_t \tag{3.5}$$

Similarly to AR models that use historical values of the target variable in a regression, a Moving Average (MA) model applies past forecast errors in a regression-style model, a moving average model of order q, denoted as MA(q), can be written as:

$$y_t = c + \epsilon_t + \phi_1 \epsilon_{t-1} + \phi_2 \epsilon_{t-2} + \cdots + \phi_q \epsilon_{t-q} \tag{3.6}$$

ARIMA combines both AR(p) and MA(q), along with the differencated target variable $y^{'}_t$ for the following stationary regression (zero mean uncorrelated errors ϵ_t with shared variance):

$$y^{'}_t = c + \epsilon_t + \sum_{l=1}^{p} \theta_l y^{'}_{t-l} + \sum_{l=2}^{q} \phi_k \epsilon_{t-k} \tag{3.7}$$

The ARIMA model family has been extensively studied, in particular in the work of George et al., 1976; Box et al., 2015b. The most popular implementation of the method ARIMA is available in the fable R package (Hyndman and Khandakar, 2008).

3.1.5 Theta Method

Another simple, efficient and well-proven method, as shown by its success in the M3 competition (Makridakis and Hibon, 2000; Assimakopoulos and Nikolopoulos, 2000). In its simplest form the Theta Method (Theta) creates a prediction using the last observation is combined with a trend function, which could be a constant, a linear or non-linear trend, or a nonparametric trend:

$$Q(\theta)_t = (1 - \frac{1}{\theta})T_{t+1} + \frac{1}{\theta}y_t \tag{3.8}$$

$$\hat{y}_{t+1} \mid {}_t = \Delta Q(\theta)_t + y_t \tag{3.9}$$

where the trend (usually linear) is denoted by T_{t+1}. The theoretical study of Theta family is performed by Nikolopoulos and Thomakos, 2019, and its extensions towards probabilistic forecasts introduced by Hyndman and Billah, 2003. The simple method was shown by Hyndman and Billah, 2003 to be equivalent to a simple exponential smoothing with drift.

Research on the topic is active, as shown by the work of Spiliotis et al., 2020a on its generalization. One of the most popular implementations of Theta baselines is available in the forecTheta library (Fiorucci and Louzada, 2020), and is based on a Theta generalization that models a short and a long term linear trend, the first is optimally estimated while the second is dynamically estimated (Fiorucci et al., 2016).

3.1.6 Conditional Heteroscedasticity Models

In some instances, like financial applications, a statistical model may exhibit an irregular pattern of variation in an error term or variable. In such cases, the assumption that the variance of the errors remains constant is no longer valid, and conditional heteroscedasticity comes into play.

This model family estimates the volatility of the target variable, usually assuming it is centered around zero[1]. The Autoregressive Conditional Heteroskedasticity (ARCH) (Engle, 1982), assumes the following autoregressive variance structure:

$$\sigma_t^2 = c + \sum_{l=1}^{p} \theta_l y_{t-l}^2 \quad \text{and} \quad y_t = \epsilon_t \sigma_t \tag{3.10}$$

The Generalized Autoregressive Conditional Heteroskedasticity (GARCH) (Bollerslev, 1986) extension assumes that the variance can be predicted by the long-term average variance, the predicted variance for the current period, and the most recent squared residual, which represents new information for the period. The three factors are weighted together in the calculation:

$$\sigma_t^2 = c + \sum_{l=1}^{p} \theta_l y_{t-l}^2 + \sum_{k=1}^{q} \phi_k \sigma_{t-k}^2 \tag{3.11}$$

[1]Under certain conditions it can be thought as an autorregresive model for the squared y_t^2 series.

3.1.7 Multiple Seasonalities

Multiple seasonal models extend on time series decomposition analysis that presents a time series as a function of other partial signals called components. The simplest form of decomposition assumes that a time series is formed by trend, seasonal component, and the remainder, where the seasonal component has cyclical patterns over time. The mathematical decomposition approach has a long history dating to the work of Buys-Ballot, 1847, to classic smoothing methods like that of Macaulay, 1931. Digitalization advancements has significantly impacted the frequency at which time series data is analyzed. In various sectors, including energy, healthcare, transportation, and telecommunications, there is a growing need to analyze time series data that exhibits multiple seasonalities or cyclical components of varying frequencies.

The field of multiple seasonalities modeling is vast and continuously expanding. In this article, we will highlight some examples and suggest referring to surveys conducted by Dokumentov and Hyndman, 2015 for more information. One modern approach is the use of regression methods, in which seasonal components are typically chosen from harmonic functions such as Seasonal-Trend regression (STR) (Dokumentov and Hyndman, 2015), or Bayesian regression, such as Facebook Prophet (Prophet)(Taylor and Letham, 2018). Another approach is iterative local polynomial projections, like Seasonal-Trend decomposition using LOESS (STL) (Cleveland et al., 1990; Bandara et al., 2021). Specialized structural models, such as double-seasonal exponential smoothing work of Taylor, 2014, are also available.

3.1.8 Markov Regime-Switching

Dynamic econometric modeling and forecasting techniques have increasingly relied on a specialized class of Markov Regime-Switching (MS) models since the late 1980s, particularly in macroeconomics and finance. These models can accommodate regime shifts and allow for temporal regime dependence, non-linearities, and is especially helpful in modeling mean reversion.

A Markov regime-switching approach represents the observed stochastic behavior y_t by K separate states or regimes, each having different underlying stochastic processes, i.e., $y_{t,k}$ of $k = 1, \ldots, K$. The switching mechanism between these states is governed by an unobserved (latent) Markov chain κ_t, which is characterized by the transition matrix $\mathbf{P} = [P(\kappa_t = i | \kappa_t = j)]$. For a simple MS example consider the conditional regression model:

$$y_{t+1} \mid t, k = \theta_{0,k} + \theta_{1,k} x_{1,t} + \cdots + \theta_{p,k} x_{p,t} + \epsilon_{t,k} \tag{3.12}$$

$$\mathbf{P} = \begin{bmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,j} & \cdots & p_{1,K} \\ p_{2,1} & p_{2,2} & \cdots & p_{2,j} & \cdots & p_{2,K} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ p_{K,1} & p_{K,2} & \cdots & p_{K,j} & \cdots & p_{K,K} \end{bmatrix} \quad \text{with} \quad \sum_{j=1}^{K} p_{i,j} = 1 \tag{3.13}$$

After the early work of Hamilton, Nelson, and Schwertz (Hamilton, 1989; Pagan and Schwert, 1990; Kim, Nelson, et al., 1999), there has been an explosion of applications and generalizations of MS-based models. Updated surveys of the method are performed by Guidolin and Pedio, 2018 and Phoong et al., 2022.

3.1.9 Vector Autorregresive Models

Forecasting tasks often involve predicting multiple related series dependent on each other. However, as a simplifying assumption, it is common to assume that these series are (conditionally) independent. While this simplification can be useful, there are cases where the benefits of modeling the series relationships outweigh the advantages of the simplified assumption.

The Vector Autoregression (VAR) time series model is often used to address this issue. This multivariate statistical model assumes that each variable is influenced not only by its past behavior but also by the past behavior of other variables in the system. By forming a set of interrelated equations:

$$\mathbf{y}_t = c + \theta_1 \mathbf{y}_{t-1} + \theta_2 \mathbf{y}_{t-2} + \cdots + \theta_p \mathbf{y}_{t-p} + \epsilon_t \tag{3.14}$$

The VAR model was introduced in 1980 (Sims, 1980) and has since been widely adopted in macroeconomics and finance, resulting in a large body of literature. Several variants and extensions have been proposed to address various challenges, including multivariate state-space (Durbin and Koopman, 2012) and MGARCH models (Bauwens et al., 2006). Researchers have also proposed alleviating over-parametrization challenges through theoretically motivated regularization (Deaton and Muellbauer, 1980) and factor analysis augmentations, such as FAVAR (Bernanke et al., 2005).

3.1.10 Count and Zero Inflated Data

Scenarios with non-Gaussian distributed target variables, such as binary, count, or zero-inflated processes, motivated the introduction of more flexible statistical frameworks, like the generalized linear model (GLM) (Nelder and Wedderburn, 1972) with Bernoulli, Poisson, and Negative binomial distributions. Other specialized distribution work includes the Zero Inflated Poisson Regression (ZIP) (Lambert, 1992), and the special classes of the exponential dispersion models (Tweedie et al., 1984; Jorgensen, 1987) that proven to be remarkably accurate in the M5 international forecasting competition (Makridakis et al., 2021).

Classic forecasting baselines like exponential smoothing are known to struggle with zero-inflated processes; The Croston Method (Croston) (Croston, 1972) improves on this, fitting two separate simple exponential smoothings to the positive values and the size zero-filled intervals. Extensions to the Croston baseline include work of Intermittent Multiple Aggregation Prediction Algorithm (IMAPA) (Syntetos and Boylan, 2021), and the Aggregate-Disaggregate Intermitent Demand Approach (ADIDA) (Nikolopoulos et al., 2011).

3.2 Machine Learning Forecasting

3.2.1 Nearest Neighbors.

Nearest-neighbor techniques rely on identifying the training set's observations that are closest in input space to regressors $\mathbf{x}$ to create predictions $\hat{y}_t(\mathbf{x})$. In particular K-Nearest Neighbor Model (KNN) defines its predictions as follows:

$$\hat{y}_t(\mathbf{x}) = \frac{1}{k} \sum_{\mathbf{x}_i \in N_k(\mathbf{x})} y_i \tag{3.15}$$

where $N_k(\mathbf{x})$, which consists of the k closest points $\mathbf{x}_i$ in the training set to $\mathbf{x}$. We then determine the prediction $\hat{y}_t(\mathbf{x})$ by averaging the responses of these k closest neighbors.

Nearest neighbor methods are unsuitable for extrapolation since they assume the underlying relationship between input features and response variable remains constant. While not commonly used in forecasting tasks, KNN can be reasonable when stationarity holds or augment the method with de-trending or differencing techniques (Martínez et al., 2019). One can replace classic Euclidean distance for specialized time series distance alternatives (Senin, 2008).

3.2.2 Support Vector Regression.

Most machine learning methods attempt to estimate functions directly from the input space, while the Support Vector Machine (SVM) takes a different approach by performing a non-linear mapping of the data into a high-dimensional space and then using simple linear functions to create linear decision boundaries in that space. SVM produces a single solution characterized by the global minimum of the optimized functional. Initially introduced in 1995 by Cortes and Vapnik, 1995 to tackle classification problems, SVM was later modified to solve regression problems (Vapnik et al., 1997; Vapnik, 1999).

The original Support Vector Regression follows the following ERM problem:

$$\begin{aligned} &\underset{w,b,\zeta_i,\zeta_i'}{\text{Minimize}} \quad C\sum_{t=1}^{T} L_\epsilon\left(y_t,\ f(x_t,w)\right) + \frac{1}{2}||w||^2 \\ &\text{where} \qquad L_\epsilon(y, f(x_t,w)) = \begin{cases} 0 \quad \text{if } |y - f(x_t,w)| \leq \epsilon \\ |y - f(x_t,w)| - \epsilon \quad \text{otherwise} \end{cases} \end{aligned} \tag{3.16}$$

SVMs were once a popular forecasting choice (Trafalis and Ince, 2000; Sapankevych and Sankar, 2009), their usage has decreased due to several factors. One of the main drawbacks of SVMs is their computational complexity (Rahimi and Recht, 2008; Williams and Seeger, 2001), especially when dealing with large datasets or high-dimensional feature spaces. Additionally, the lack of interpretability in SVMs can be challenging as their non-linear transformations can be difficult to understand and often require complex kernel functions. Furthermore, unlike neural networks, SVMs require substantial feature engineering effort to compete in many applications.

3.2.3 Gaussian Processes.

Similar to SVMs, Gaussian Process (GP) methods also utilize kernel functions to model the covariance between data points (Williams and Rasmussen, 2006). In the case of GPs, kernel functions are used to model the covariance of a continuous stochastic process. Although GP's interpolation is are particularly suitable for time-series signal decomposition (known as Kriging) it can also be applied to long-horizon forecasting and when uncertainty estimation is important (Roberts et al., 2013).

3.2.4 Decision Trees.

Decision trees are a versatile class of regression methods that have gained widespread adoption, some popular variants include Random Forests (RF) (Breiman, 1996; Breiman, 2001) and Gradient Boosting Decision Trees (GBD) (Freund and Schapire, 1997; Chen and Guestrin, 2016; Ke et al., 2017). These models have consistently performed well in forecasting competitions such as Kaggle (Bojer and Meldgaard, 2021), M4 (Makridakis et al., 2020a), and M5 (Makridakis et al., 2021), and have become well-established baselines that excel under time constraints.

Tree-based methods are often the selection for users seeking effective, easy-to-use black box learners for forecasting tasks (Januschowski et al., 2022). Their success in forecasting competitions can be attributed to several factors:

- These method's mature software implementations have garnered strong community support, making them easily accessible and widely adopted.
- These highly robust methods can effectively handle noise, missing data, and variations in time series scales, allowing for reliable predictions even in challenging datasets.
- Unlike other methods, Tree-based methods do not require extensive model tuning or parameter adjustments to achieve competitive performance, making them an attractive option for users without specialized knowledge of machine learning.

It is important to note that Tree-based methods generally partition the training input space into regions and assign a specific output value to each region. However, when presented with inputs outside the training set's range, decision trees often fail to predict accurately; the decision tree may assign the inputs to a region with an edge or default output value. For this reason, the methods are not ideal for extrapolation or non-stationary settings. This is a well-known problem and research on this area is an active topic.

In contrast, other machine learning models, such as neural networks and support vector machines, can learn continuous and smooth functions that can be used for extrapolation, making them better suited to handle situations with values outside the training set's range.

3.3 Neural Forecasting

In recent years, there has been an increasing need to forecast large numbers of related time series rather than just a few individual ones. In such scenarios, global models can use data from collections of related time series to learn complex relationships without the risk of over-fitting. In addition to improving accuracy, this approach can save time and effort by eliminating the need for experts to select and prepare covariates and models, as traditional techniques require (Wen et al., 2017; Smyl, 2019; Semenoglou et al., 2021). Before the advent of deep learning, it was necessary to design complex pipelines involving clustering similar series, creating specific features for each series, like each series' special handling of promotional or holiday effects, and selecting different model specifications based on individual characteristics.

Neural forecasting provides a simpler solution. These models require only a small amount of data preprocessing before they can learn an end-to-end solution to the forecasting problem. In particular, data processing is included in the model and optimized jointly to produce the best possible forecast. Unlike traditional pipelines, which rely on heuristics such as expert-designed components and manual covariate design, deep learning forecasting pipelines rely almost entirely on what the model can learn from the data.

Deep Learning's ability to learn the relationship between features and forecasts and the features' representation itself is called representation learning. It enables it to automatically discover good representations from raw data without relying on manual feature engineering (Goodfellow et al., 2016). The ability depends on introducing data representations constructed on top of other simpler representations obtained through the layers of the models. Representation learning has solid theoretical foundations, including the *Universal Approximation Theorem* (Hornik, 1991; Cybenko, 1989). The theorem states that even a simple architecture like a feedforward neural network can arbitrarily approximate any continuous function, conditioned on its depth and number of hidden units.

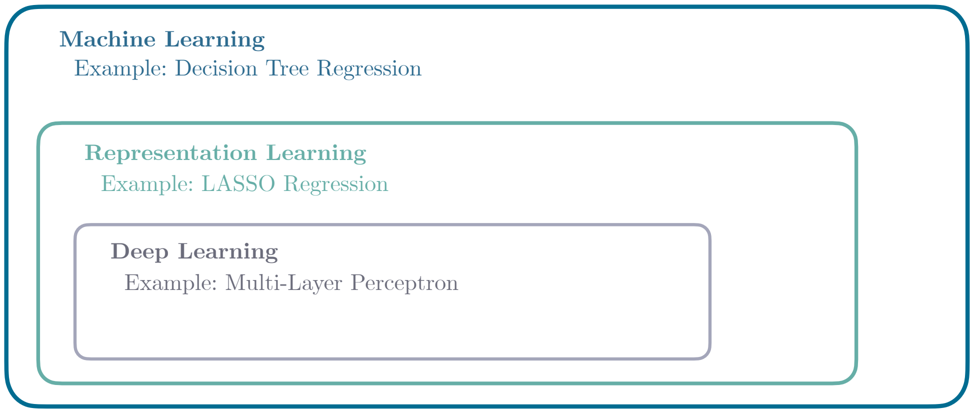

Machine Learning
Example: Decision Tree Regression
Representation Learning
Example: LASSO Regression
Deep Learning
Example: Multi-Layer Perceptron

Figure 3.2: Representation learning is a powerful machine learning tool, as it allows us to automatically discover and extract useful features from the data without manual feature engineering.

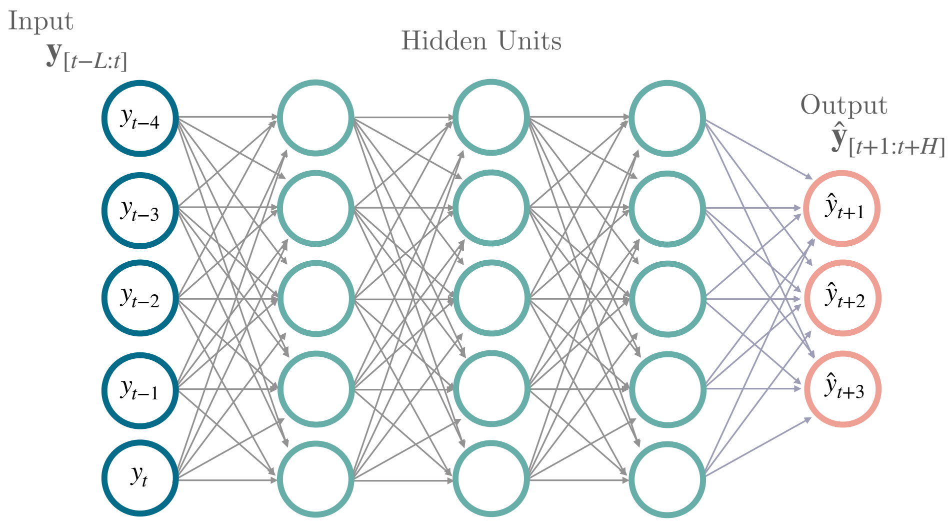


Figure 3.3: When the relationship between regressors $\mathbf{y}_{[t-L:t]}$ and forecasts $\hat{\mathbf{y}}_{[t+1:t+H]}$ is complex, preparing covariates and selecting models is time-consuming. Neural forecasting solves this difficulty through representation learning, decomposing complex functions into a series of simpler ones, each function represented by the model's different layers. The input layer processes the original features, while subsequent hidden layers extract increasingly abstract features. These hidden features are not initially in the data; the model determines which transformations are most useful for forecasting. Overall, deep learning simplifies learning the relationship between regressors and forecasts.

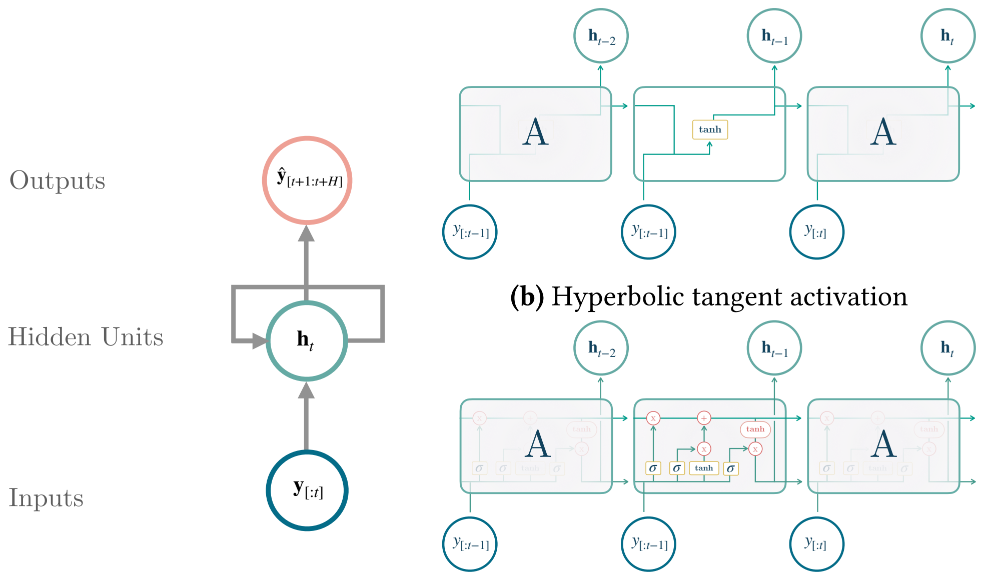
3.3.1 Architecture Building Blocks

There are many different neural network architectures, and their strengths and weaknesses make them suited for different tasks. Researchers and practitioners in deep learning are constantly seeking to improve model performance, reduce computational costs, and expand their range of applications. The following subsections will briefly cover their most important building blocks.

Multi Layer Perceptron

The simplest network architecture is the Multi Layer Perceptron (MLP) (Rosenblatt, 1961). It operates as an autoencoder since its initial layers operate as an encoder function that converts the raw inputs into a different representation and a decoder function that converts the representation into the desired output. In the edge case where the autoencoder has a single layer, the autoencoder turns into a linear Autorregresive Models (AR) model.

The feedforward neural network gets its name because inputs are processed in one direction, flowing through the network to produce outputs. In Figure 3.3, each circle represents a node in the network, while the edges depict the transformations applied by the network. A layer is a set of nodes that use an affine transformation followed by a nonlinear activation function. Activation functions serve the purpose of adapting the output of the network to specific domains, for example, by predicting distribution parameters.



(a) Single Layer RNN

(b) Hyperbolic tangent activation

(c) LSTM activation

Figure 3.4: (a) Recurrent Neural Networks are a family of models specialized on sequential data. Their internal mechanism allows them to deal with inputs of varying length as its weights are shared across time steps. (b) In its original versions RNNS would use hyperbolic tanget activations. (c) To tackle the exploding and vanishing gradient problems new activations have been proposed.

Recurrent Neural Networks

One significant limitation of Multilayer Perceptrons (MLPs) is their over parametrization. Since the model is fully connected, it does not leverage any specific structure in the data, as other architectures do. Additionally, its fixed parameters limit their ability to process inputs of varying lengths. However, most modern neural networks use MLPs as an essential building block.

Recurrent Neural Network (RNN)s (Rumelhart et al., 1986; Elman, 1990) are a model family specialized on sequential data. In contrast to MLPs that are static architectures, RNNs are dynamic and can handle sequences of varying lengths because they have a mechanism for updating their internal state based on the entire sequence history. The mechanism's weights are shared across all time steps. RNNs dynamic memory depends feeds its hidden units back to themselves at each time step, and endows it with the ability to capture long-term dependencies.

While training RNNs over long sequences using back-propagation is that their gradients tend to vanish or explode. Long Short Term Memory (LSTM) (Gers et al., 2000; Sak et al., 2014) introduced a clever mitigation through its activation's gating mechanism that selectively learns which information to keep and which to discard from previous time steps in the input sequence.

Additional computational efficiency improvements include the Gated Recurrent Unit (GRU) that simplifies LSTM's activation by combining its gating mechanism (Chung et al., 2014; Cho et al., 2014). Still due to the sequential processing nature of RNNs, their operations cannot be parallelized in contrast with those of MLP-based and Transformer-based architectures, for this reason they have often worse computational efficiency.

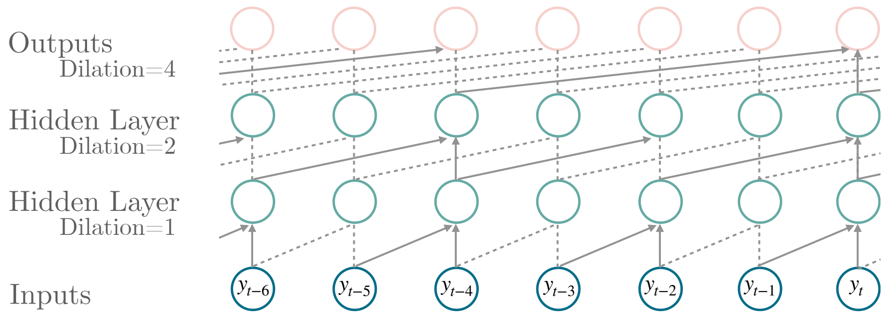


Figure 3.5: Simple convolutional architectures can outperform modern RNNs. By skipping temporal connections, causal convolution filters efficiently model longer memory. Stacking multiple convolutional layers creates higher-order features for accurate forecasts.

Convolutional Neural Networks

The Convolutional Neural Networks (CNN) (LeCun et al., 1989) constitue a specialized family of models designed for inputs with a known ordinal structure, such as images and time series. Unlike MLPs, CNNs utilize convolutional layers that employ filters operating locally on subsets of the input data, which are shared across the input range. For time series forecasting, Temporal Convolution Network (TCN) (Oord et al., 2016; Bai et al., 2018) performs a moving weighted sum by sliding a filter across the input data. The weight sharing of the filter drastically reduces the number of free parameters compared to the dense layers of MLPs and improving generalization.

Causal convolution filters can be applied to larger time spans while remaining computationally efficient by skipping temporal connections (Chang et al., 2017). This approach, depicted in Figure 3.5, can be further extended by stacking multiple convolutional layers on top of each other, which combines low-level features extracted in earlier layers to derive higher-order features. This process of feature extraction allows for the creation of more complex representations of the input data, enabling more accurate predictions or classifications.

Attention Mechanism and Transformer Architectures

The attention mechanism overcomes the information bottleneck in Sequence to Sequence architectures by selectively focusing on relevant parts of the input sequence. By weighting the importance of different input elements based on the decoder's current state, attention allows the model to attend to the most relevant parts (Chorowski et al., 2014; Bahdanau et al., 2016).

The Transformer (Transformer) (Vaswani et al., 2017) architecture is a feedforward network that processes the input sequence in parallel rather than sequentially like RNNs. The Transformer uses multi-headed attention to capture the relationships between different input sequence elements. While Transformer-based models have achieved great success in natural language processing and computer vision tasks, their adaptation for forecasting purposes is a relatively recent area of research that requires further development and refinement to reach comparable levels of success. As we show in Chapter 6, early attempts have been partially unsuccessful (Zhou et al., 2020; Wu et al., 2021; Zhou et al., 2022).

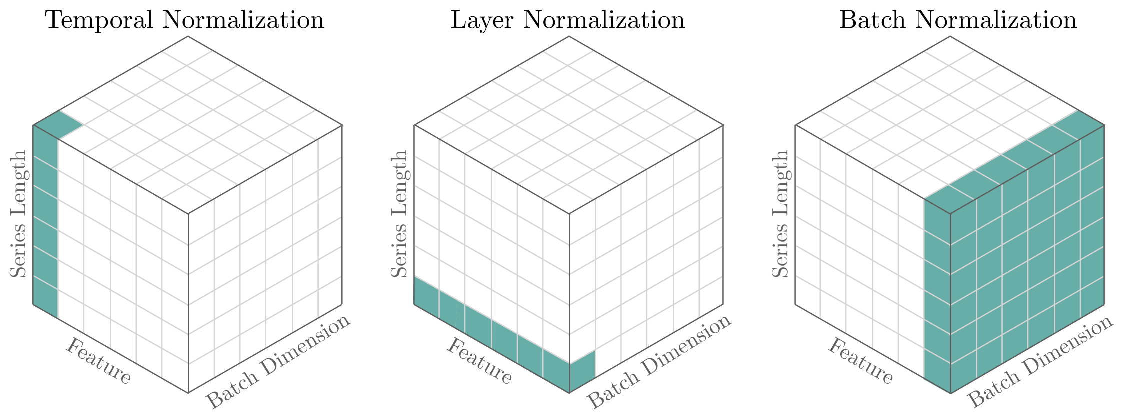


Figure 3.6: Temporal normalization (left), layer normalization (center) and batch normalization (right). The entries in green show the components used to compute the normalizing statistics.

Temporal Normalization

Temporal normalization has proven essential in neural forecasting tasks, enabling the network's non-linearities to express themselves. Forecasting scaling methods take particular interest in the temporal dimension where most of the variance dwells, contrary to other deep learning techniques like BatchNorm, which normalizes across batch and temporal dimensions, and LayerNorm, which normalizes across the feature dimension.

3.3.2 Neural Forecasting Architectures

We reviewed the fundamental neural network's building blocks in the preceding subsections. Although the neural network architectures for modern applications have become more intricate, they still consist of a fusion of elementary structures, such as MLPs, RNNs, CNNs, and attention mechanisms. Here we briefly review and point to notable or well-performing architectures.

The Sequence to Sequence Architecture (Seq2Seq) (Graves, 2013) was a breakthrough in sequential data processing tasks due to its ability to handle variable-length input and output sequences. Variants of the architecture have been applied to large industrial forecasting systems, such as Deep Auto Regressive Network (DeepAR) (Salinas et al., 2020), Multi Quantile Forecaster Family (MQForecaster) (Wen et al., 2017; Eisenach et al., 2021) and Temporal Fusion Transformer (TFT) (Lim et al., 2021), with great effect. Neural forecasting has transcended industry boundaries into academia due to its outstanding performance in the latest forecasting competitions (Makridakis et al., 2018c; Makridakis et al., 2020b). The latest submissions witnessed the Exponential Smoothing Recurrent Neural Network (ESRNN) (Smyl, 2019) and Neural Basis Expansion Analysis (NBEATS) (Oreshkin et al., 2020) obtain first and third place, respectively.

Cross-learning is common to all these architectures, it involves optimizing a shared model across a collection of related time series. The clever integration of temporal normalization strategies within the architectures has supported this approach (Semenoglou et al., 2021).

Part II

Contributions

In the computer field,
the moment of truth is a running program; all else is prophecy.

Herbert Simon

We present the thesis's main contributions in three case studies that combine neural forecasting methods with econometric and statistical inspirations. Chapter 4 introduces NBEATSx, which extends neural basis expansion analysis with exogenous variables improving its accuracy and providing interpretable signal decomposition capabilities. In Chapter 5, we introduce a novel probabilistic mixture model to tackle the hierarchical forecasting task. The new method extends network's capabilities to arbitrarily approximate functions to approximate distributions, including those with coherence constraints. In Chapter 6, we tackle long-horizon forecasting with NHITS, a Wavelet analysis-inspired approach that specializes multi-step forecasting strategy into different frequencies through time.

Chapter 4

Interpretable Neural Forecasting

4.1 Summary

Existing neural forecasting approaches can be limited in their interpretability, so we created NBEATSx to address this challenge. NBEATSx extends the neural basis expansion analysis method by incorporating exogenous variables, significantly improving its accuracy and enabling the integration of multiple sources of helpful information. The neural network used in NBEATSx provides an interpretable signal decomposition, allowing users to visualize the relative impact of trend and seasonal components and the interactions with exogenous factors. NBEATSx makes it easier to understand how the model composes its predictions.

4.2 Motivation

In the last decade, a significant progress has been made in the application of deep learning to forecasting tasks, with models such as the Exponential Smoothing Recurrent Neural Network (ESRNN) (Smyl, 2019) and the Neural Basis Expansion Analysis (NBEATS) (Oreshkin et al., 2020), outperforming classical statistical approaches in the recent M4 competition (Makridakis et al., 2020a). Despite this success we still identify two possible improvements, namely the integration of time-dependent exogenous variables as their inputs and the interpretability of the neural network outputs. Neural networks have proven powerful and flexible, yet there are several situations where our understanding of the model's predictions can be as crucial as their accuracy, which constitutes a barrier for their wider adoption. The interpretability of the algorithm's outputs is critical because it encourages trust in its predictions, improves our knowledge of the modeled processes, and provides insights that can improve the method itself.

Additionally, the absence of time-dependent covariates makes these powerful models unsuitable for many applications. For instance, Electricity Price Forecasting (EPF) is a task where covariate features are fundamental to obtain accurate predictions. For this reason, we chose this challenging application as a test ground for our proposed forecasting methods.

In this chapter, we address the two mentioned limitations by first extending the neural basis expansion analysis, allowing it to incorporate temporal and static exogenous variables. And second, by further exploring the interpretable configuration of NBEATS and showing its use as a time-series signal decomposition tool. We refer to the new method as NBEATSx. The main contributions of this paper include:

(i) **Incorporation of Exogenous Variables:** We propose improvements to the NBEATS model to incorporate time dependent as well as static exogenous variables. For this purpose, we have designed a special substructure built with convolutions, to clean and encode useful information from these covariates, while respecting time dependencies present in the data. These enhancements greatly improve the accuracy of the NBEATS method, and extend its interpretability capabilities, so rare in neural forecasting.

(ii) **Interpretable Time Series Signal Decomposition:** Our method combines the power of non-linear transformations provided by neural networks with the flexibility to model multiple seasonalities and simultaneously account for interaction events such as holidays and other covariates, all while remaining interpretable. The extended NBEATSx architecture allows to decompose its predictions into the classic set of level, trend, and seasonality, and identify the effects of exogenous covariates.

(iii) **Electricity Price Forecasting Comparison:** We showcase the use of NBEATSx model on five EPF tasks achieving state-of-the-art performance on all of the considered datasets. We obtain accuracy improvements of almost 20% in comparison to the original NBEATS and ESRNN architectures, and up to 5% over other well-established machine learning, EPF-tailored methods (Lago et al., 2021a).

4.3 Related Work

4.3.1 Electricity Price Forecasting

The Electricity Price Forecasting (EPF) task aims at predicting the spot (balancing, intraday, day-ahead) and forward prices in wholesale markets. Since the workhorse of short-term power trading is the day-ahead market with its conducted once-per-day uniform-price auction (Mayer and Trück, 2018), the vast majority of research has focused on predicting electricity prices for the 24 hours of the next day, either in a point (Weron, 2014; Lago et al., 2021a) or a probabilistic setting (Nowotarski and Weron, 2018). There also are studies on EPF for very short-term (Narajewski and Ziel, 2020), as well as mid- and long-term horizons (Ziel and Steinert, 2018a). The recent expansion of renewable energy generation and large-scale battery storage has induced complex dynamics to the already volatile electricity spot prices, turning the field into a prolific subject on which to test novel forecasting ideas and trading strategies (Chitsaz et al., 2018; Gianfreda et al., 2020; Uniejewski and Weron, 2021).

Energy markets' liberalization and renewable energy sources induced complex dynamics and volatility to electricity prices (Angelica Gianfreda and Pelagatti, 2016; Gianfreda et al., 2020; Muniain and Ziel, 2020), turning them into a prolific subject on which to test forecasting ideas.

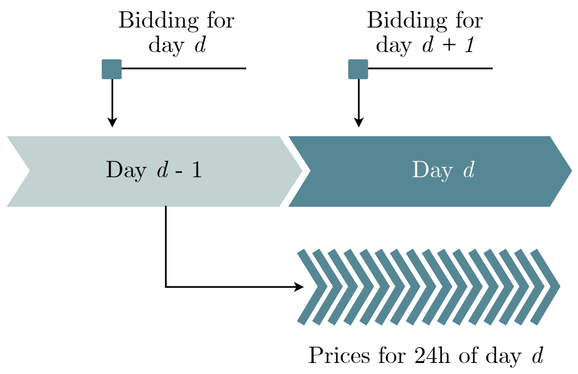


Figure 4.1: The *day-ahead* auction market allows participants to purchase and sell electric energy at prices determined on day $d - 1$ for the next day d. The market establishes the 24 hourly prices simultaneously, and it is usual that the prices are published at midday.

Out of the numerous approaches to EPF developed over the last two decades, two classes of models are of particular importance when predicting day-ahead prices – statistical (also called econometric or technical analysis), in most cases based on linear regression, and computational intelligence (also referred to as artificial intelligence, non-linear or machine learning), with neural networks being the fundamental building block. Among the latter, many of the recently proposed methods utilize deep learning (Wang et al. 2017; Lago et al. 2018a; Marcjasz 2020), or are hybrid solutions, that typically comprise data decomposition, feature selection, clustering, forecast averaging and/or heuristic optimization to estimate the model (hyper)parameters (Nazar et al., 2018; Li and Becker, 2021).

Unfortunately, as argued by Lago et al., 2021a, the majority of the neural network EPF related research suffers from too short and limited to a single market test periods, lack of well performing and established benchmark methods, and/or incomplete descriptions of the pipeline and training methodology resulting in poor reproducibility. To address these shortcomings, our models are compared across two-year out-of-sample periods from five power markets and using two highly competitive benchmarks recommended in previous studies: the *Lasso Estimated Auto-Regressive* (LEAR) model and a (relatively) parsimonious *Deep Neural Network* (DNN).

4.4 Methodology

4.4.1 Neural Basis Expansion Analysis

The NBEATSx model offers a solution to the multivariate regression problem

$$\mathrm{P}(\mathbf{y}_{[t:t+H]} \mid \mathbf{y}_{[:t]},\ \mathbf{X}_{[:t+H]}) := \mathrm{P}(\mathbf{y}_{[t+1:t+H]} \mid \mathbf{y}_{[t-L:\,t]},\ \mathbf{X}_{[t-L:\,t+H]}) \tag{4.1}$$

where $\mathbf{y}_{[t:t+h]}, \mathbf{y}_{[:t]}, \mathbf{X}_{[:t+H]}$ represent future and past observations of the target time series up until time t, and the the exogenous variables available at the prediction time, respectively. And L denotes the number of lags considered in the regression, and H is the forecast horizon.

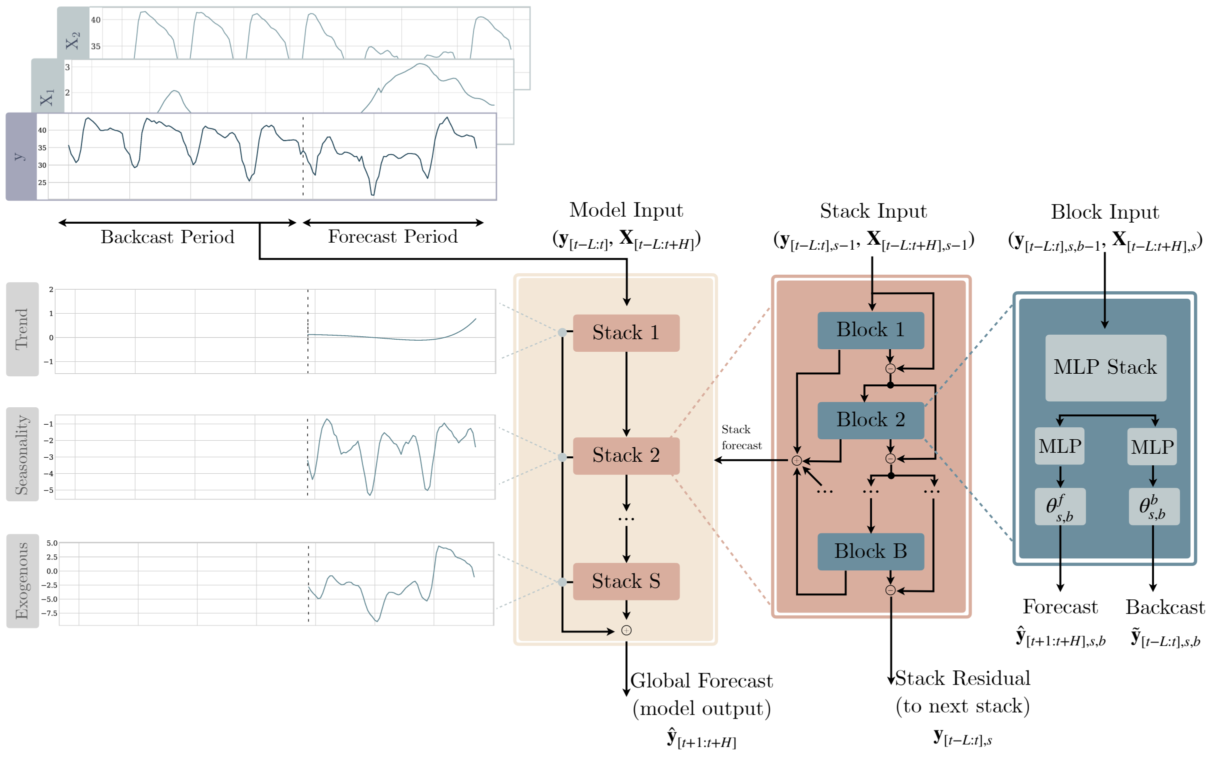


Figure 4.2: The Building blocks of the NBEATSx are structured as a system of multilayer perceptrons with ReLU based nonlinearities. Blocks overlap using the doubly residual stacking principle for the backcast $\tilde{\mathbf{y}}_{[t-L:t],s,b}$ and forecast $\hat{\mathbf{y}}_{[t+1:t+H],s,b}$ outputs of the b-th block within the s-th stack. The final predictions $\hat{\mathbf{y}}_{[t+1:t+H]}$ are composed by aggregating the outputs of the stacks.

The NBEATSx framework decomposes the objective signal by performing separate local nonlinear projections of the data onto basis functions across its different blocks. Figure 4.2 depicts the general architecture of the model. Each block consists of a Multi Layer Perceptron (MLP) (Rosenblatt, 1961) which learns expansion coefficients for the backcast and forecast elements. The backcast model is used to clean the inputs of subsequent blocks, while the forecasts are summed to compose the final prediction. The blocks are grouped in stacks. Each of the potentially multiple stacks specializes in a different variant of basis functions.

To continue NBEATSx' description, we introduce the following notation: the objective signal is represented by the vector $\mathbf{y}$, the inputs for the model are the backcast window vector $\mathbf{y}_{[t-L:t]}$ of length L, and the forecast window vector $\mathbf{y}_{[t+1:t+H]}$ of length H; where L denotes the length of the lags available as classic autoregressive features, and H is the forecast horizon treated as the objective. While NBEATS only admits as regressor the backcast period of the target variable $\mathbf{y}_{[t-L:t]}$, the NBEATSx incorporates covariates in its analysis denoted with the matrix $\mathbf{X}$. Figure 4.2 shows an example where the target variable is the hourly electricity price, the backcast vector has a length L of 96 hours, and the forecast horizon H is 72 hours, in the example, the covariate matrix $\mathbf{X}$ is composed of wind power production and electricity load. For the EPF comparative analysis of Section 4.5.6 the horizon considered is $H = 24$ that corresponds to day-ahead predictions, while backcast inputs $L = 168$ correspond to a week of lagged values.

For its predictions, the NBEATSx receives a local vector of inputs corresponding to the backcast period, making the computations exceptionally fast. The model can still represent longer time dependencies through its local inputs from the exogenous variables; for example, it can learn long seasonal effects from calendar variables. As shown in Figure 4.2, the NBEATSx is composed of S stacks of B blocks each, the input $\mathbf{y}_{[t-L:t]}, \mathbf{X}_{[t-L:t]}$ of the first block consists of L lags of the target time series $\mathbf{y}$ and the exogenous matrix $\mathbf{X}$, while the inputs of each of the subsequent blocks include residual connections with the backcast output of the previous block. We will describe in detail in the next subsections the blocks, stacks, and model predictions.

4.4.2 Blocks

For a given s-th stack and b-th block within it, the NBEATSx model performs two transformations, depicted in the blue rectangle of Figure 4.2. The first transformation, defined in Equation (4.2) and Equation (4.3), takes the input data $(\mathbf{y}_{[t-L:t],s,b-1}, \mathbf{X}_{[t-L:t+H],s})$, and applies a *Fully Connected Multi Layer Perceptron* (MLP; Rosenblatt 1961) to learn hidden units $\mathbf{h}_{s,b} \in \mathbb{R}^{N_h}$ that are linearly adapted into the forecast $\theta^{f}_{s,b} \in \mathbb{R}^{N_{st}}$ and backcast $\theta^{b}_{s,b} \in \mathbb{R}^{N_{st}}$ expansion coefficients, with N_{st} the dimension of the stack basis.

$$\mathbf{h}_{s,b} = \mathbf{MLP}_{s,b}\left(\mathbf{y}_{[t-L:t],s,b}\right) \quad \text{or} \quad \mathbf{h}_{s,b} = \mathbf{MLP}_{s,b}\left(\mathbf{y}_{[t-L:t],s,b}, \mathbf{X}_{[t-L:t+H],s}\right) \tag{4.2}$$

$$\theta^{b}_{s,b} = \mathbf{LINEAR}^{b}\left(\mathbf{h}_{s,b}\right) \quad \text{and} \quad \theta^{f}_{s,b} = \mathbf{LINEAR}^{f}\left(\mathbf{h}_{s,b}\right) \tag{4.3}$$

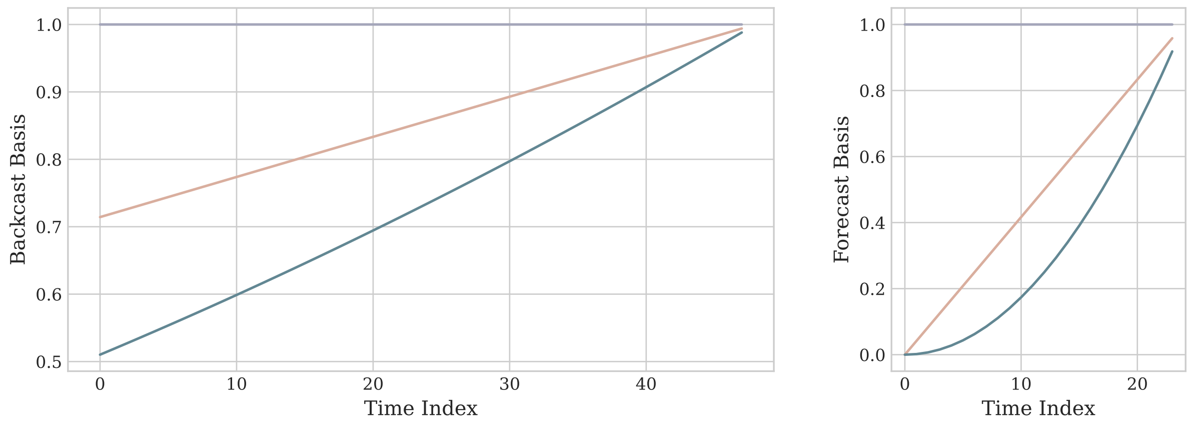
The second transformation, in Equation (4.4), consists of a basis expansion between the learnt coefficients and the block's basis $\mathbf{V}_{[t-L:t],s,b} \in \mathbb{R}^{L \times N_{st}}$ and $\mathbf{V}_{[t+1:t+H],s,b} \in \mathbb{R}^{H \times N_{st}}$, this transformation results in the backcast $\tilde{\mathbf{y}}_{[t-L:t],s,b}$ and forecast $\hat{\mathbf{y}}_{[t+1:t+H],s,b}$ components.

$$\tilde{\mathbf{y}}_{[t-L:t],s,b} = \mathbf{V}_{[t-L:t],s,b}\,\theta^{b}_{s,b} \quad \text{and} \quad \hat{\mathbf{y}}_{[t+1:t+H],s,b} = \mathbf{V}_{[t+1:t+H],s,b}\,\theta^{f}_{s,b} \tag{4.4}$$

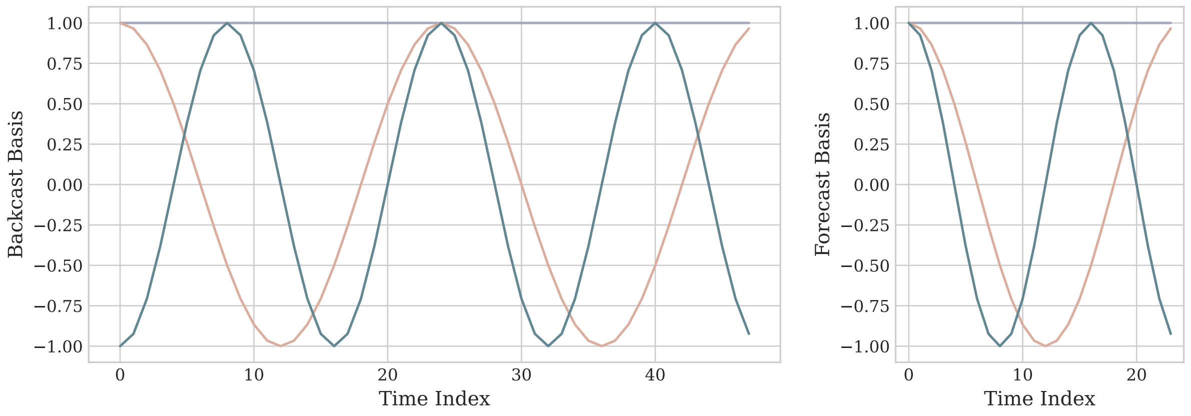
4.4.3 Stacks and Residual Connections

The blocks are organized into stacks using the doubly residual stacking principle, which is described in Equation (4.5) and depicted in the brown rectangle of Figure 4.2. The residual backcast $\mathbf{y}_{[t-L:t],s,b+1} \in \mathbb{R}^{L}$ allows the model to subtract the component associated to the basis of the s, b-th stack and block $\mathbf{V}_{[t-L:t]s,b}$ from $\mathbf{y}_{[t-L:t]}$, which can be also thought of as a sequential decomposition of the modeled signal. In turn, this methodology helps with the optimization procedure as it prepares the inputs of the subsequent layer making the downstream forecast easier. The stack forecast $\hat{\mathbf{y}}_{[t+1:t+H],s} \in \mathbb{R}^{H}$ aggregates the partial forecasts from each block.

$$\mathbf{y}_{[t-L:t],s,b+1} = \mathbf{y}_{[t-L:t],s,b} - \tilde{\mathbf{y}}_{[t-L:t],s,b} \qquad \text{and} \qquad \hat{\mathbf{y}}_{[t+1:t+H],s} = \sum_{b=1}^{B} \hat{\mathbf{y}}_{[t+1:t+H],s,b} \tag{4.5}$$



(a) Trend Basis



(b) Harmonic Basis

Figure 4.3: Examples of polynomial and harmonic basis in the interpretable configuration of the neural basis expansion analysis. The slowly varying basis allow NBEATS to model trends and seasonalities.

4.4.4 Model predictions

The final predictions $\hat{\mathbf{y}}_{[t+1:t+H]} \in \mathbb{R}^H$, we later denote particular day predictions as $\hat{\mathbf{y}}_d$, shown in the yellow rectangle of Figure 4.2, are obtained by summation of all the stack predictions.

$$\hat{\mathbf{y}}_{[t+1:t+H]} = \sum_{s=1}^{S} \hat{\mathbf{y}}_{[t+1:t+H],s} \tag{4.6}$$

The additive generation of the forecast implies a very intuitive decomposition of the prediction components when the bases within the blocks are interpretable.

4.4.5 NBEATSx Configurations

The original *neural basis expansion analysis* method proposed two configurations based on the assumptions encoded in the learning algorithm by selecting the basis vectors $\mathbf{V}_{[t-L:t],s,b}$ and $\mathbf{V}_{[t+1:t+H],s,b}$ used in the blocks from Equation (4.4). A mindful selection of restrictions to the basis allows the model to output an interpretable decomposition of the forecasts, while allowing the basis to be freely determined can produce more flexible forecasts by effectively removing any constraints on the form of the basis functions. In this subsection, we present both interpretable and generic configurations, explaining in particular how we propose to include the covariates in each case. We limit ourselves to the analysis of the forecast basis, as the backcast basis analysis is almost identical, only differing by its extension over time. We show an example in Figure 4.3.

Interpretable Configuration

The choice of basis vectors relies on time series decomposition techniques that are often used to understand the structure of a given time series and patterns of its variation. Work in this area ranges from classical smoothing methods (Macaulay, 1931) and their extensions such as X-11-ARIMA, (Shishkin et al., 1967; Dagum, 1980), X-12-ARIMA, (Findley et al., 1998), and X-13-ARIMA-SEATS, (U.S. Census Bureau, 2013), to modern approaches such as TBATS (Livera et al., 2011), and STR/STL (Cleveland et al., 1990; Dokumentov and Hyndman, 2015). To encourage interpretability, the blocks within each stack may use harmonic functions, polynomial trends, and exogenous variables directly to perform their projections. The partial forecasts of the interpretable configuration are described by Equation (4.7)-Equation (4.9).

$$\hat{\mathbf{y}}^{trend}_{[t+1:t+H],s,b} = \sum_{i=0}^{N_{pol}} \mathbf{t}^i\, \theta^{trend}_{s,b,i} \equiv \mathbf{T}\, \theta^{trend}_{s,b} \tag{4.7}$$

$$\hat{\mathbf{y}}^{seas}_{[t+1:t+H],s,b} = \sum_{i=0}^{N_{har}} \cos\left(2\pi i\mathbf{t}\right) \theta^{seas}_{s,b,i} + \sin\left(2\pi i\mathbf{t}\right) \theta^{seas}_{s,b,i+\lfloor H/2\rfloor} \equiv \mathbf{S}\, \theta^{seas}_{s,b} \tag{4.8}$$

$$\hat{\mathbf{y}}^{exog}_{[t+1:t+H],s,b} = \sum_{i=0}^{N_{x}} \mathbf{X}_{[t+1:t+H],i}\, \theta^{exog}_{s,b,i} \equiv \mathbf{X}_{[t+1:t+H]}\, \theta^{exog}_{s,b} \tag{4.9}$$

where the time vector $\mathbf{t}^\intercal = [0, 1, 2, \ldots, H-2, H-1]/H$ is defined discretely. When the basis $\mathbf{V}_{[t+1:t+H],s,b}$ is $\mathbf{T} = [\mathbf{1}, \mathbf{t}, \ldots, \mathbf{t}^{N_{pol}}] \in \mathbb{R}^{H\times(N_{pol}+1)}$, where N_{pol} is the maximum polynomial degree, the coefficients are those of a polynomial model for the trend. When $\mathbf{V}_{[t+1:t+H],s,b}$ are harmonic $\mathbf{S} = [\mathbf{1}, \cos(2\pi\mathbf{t}), \ldots, \cos(2\pi i\mathbf{t}), \sin(2\pi\mathbf{t}), \ldots, \sin(2\pi i\mathbf{t})] \in \mathbb{R}^{H\times N_{har}}$, and $N_{har} = (H-1)$, the coefficients vector $\theta^{f}_{s,b}$ can be interpreted as Fourier transform coefficients. The exogenous basis expansion can be thought as a time-varying local regression when the basis is the matrix $\mathbf{X}_{[t+1:t+H]} = [\mathbf{X}_1, \ldots, \mathbf{X}_{N_x}] \in \mathbb{R}^{H\times N_x}$, where N_x is the number of exogenous variables. The resulting models can flexibly reflect common structural assumptions, in particular using the interpretable bases, as well as their combinations.

In this paper, we propose including one more type of stack to specifically represent exogenous variable basis as described in Equation (4.9) and depicted in Figure 4.2. In the original NBEATS framework (Oreshkin et al., 2020), the interpretable configuration usually consists of a trend stack followed by a seasonality stack, each containing three blocks. Our NBEATSx extension of this configuration consists of three stacks, one of each type of factors (trend, seasonal, exogenous). We refer to this interpretable and its enhanced interpretable configuration as the NBEATS-I and NBEATSx-I models, respectively.

Generic Configuration

For the generic configuration, the basis of the non linear projection in Equation (4.4) corresponds to canonical vectors, that is $\mathbf{V}_{[t+1:t+H],s,b} = I_{H\times H}$, an identity matrix of dimensionality equal to the forecast horizon H that matches the coefficient's cardinality $|\theta^{f}_{s,b}| = H$.

$$\hat{\mathbf{y}}_{[t+1:t+H],s,b} = \mathbf{V}_{[t+1:t+H],s,b}\,\theta^{f}_{s,b} = \theta^{f}_{s,b} \tag{4.10}$$

This basis enables NBEATSx to effectively behave like a classic *Fully Connected Multilayer Perceptron* (MLP). The output layer of the MLP inside each block has H neurons, that correspond to the forecast horizon, each producing the forecast for one particular time point of the forecast period. This can be understood as the basis vectors being learned during optimization, allowing the waveform of the basis of each stack to be freely determined in a data-driven fashion. Compared to the interpretable counterpart described in Section 4.4.5, the constraints on the form of the basis functions are removed. This affords the generic variant more flexibility and power at representing complex data, but it can also lead to less interpretable outcomes and potentially escalated risk of overfitting.

For the NBEATSx model with the generic configuration, we propose a new type of exogenous block that encodes the past $\mathbf{C}_{[t+1:t+H],s,b} \in \mathbb{R}^{H\times N_f}$ from the time-dependent covariates with an *encoder* convolutional sub-structure:

$$\begin{aligned} \hat{\mathbf{y}}^{exog}_{[t+1:t+H],s,b} &= \sum_{i=1}^{N_f} C_{[t+1:t+H],s,b,i}\,\theta^{f}_{s,b,i} \equiv \mathbf{C}_{[t+1:t+H],s,b}\,\theta^{f}_{s,b} \\ \text{with} \qquad \mathbf{C}_{[t+1:t+H],s,b} &= \mathrm{TCN}(\mathbf{X}_{[t-L:t+H],s})_{[t+1:t+H]} \end{aligned} \tag{4.11}$$

In the previous equation, a Temporal Convolution Network (TCN) (Bai et al., 2018; Oord et al., 2016) is employed as an *encoder*, but any neural network with a sequential structure will be compatible with the backcast and forecast branches of the model, and could be used as an *encoder*. Temporal convolutions can be an effective alternative to RNNs as it is also able to capture long term dependencies and interactions of covariates by stacking multiple layers, while dilations help it keep the models computationally tractable. In addition, convolutions have a very convenient interpretation as a weighted moving average signal filters. The final linear projection and the additive composition of the predictions can be interpreted as a *decoder*. The original NBEATS configuration includes only one generic stack with dozens of blocks, while our proposed model includes both the generic and exogenous stacks, with the order determined via hyperparameter tuning. We refer to this configuration as the NBEATSx-G model.

Table 4.1: Datasets used in our empirical study. For the five day-ahead electricity markets considered, we report the test period dates and two influential covariate variables.

Market	Exogenous Variable 1	Exogenous Variable 2	Test Period
NP	day-ahead load	day-ahead wind generation	27-12-2016 to 24-12-2018
PJM	2 day-ahead system load	2 day-ahead COMED load	27-12-2016 to 24-12-2018
EPEX-FR	day-ahead load	day-ahead total France generation	04-01-2015 to 31-12-2016
EPEX-BE	day-ahead load	day-ahead total France generation	04-01-2015 to 31-12-2016
EPEX-DE	day-ahead zonal load	day-ahead wind and solar generation	04-01-2016 to 31-12-2017

4.5 Experiments

4.5.1 Electricity Price Datasets

In the short-term electricity price forecasting tasks the objective is to predict day-ahead prices. Five major power markets are used in the empirical evaluation, all comprised of hourly observations of the prices and two influential temporal exogenous variables that extend for 2,184 days (312 weeks, six years). From the six years of available data for each market, we hold two years out, to test the forecasting performance of the algorithms. The length and diversity of the test sets allow us to obtain accurate and highly comprehensive measurements of the robustness and the generalization capabilities of the models.

Table 4.1 summarizes the key characteristics of each market. The Nord Pool electricity market (NP), which corresponds to the Nordic countries exchange, contains the hourly prices and day-ahead forecasts of load and wind generation. The second dataset is the Pennsylvania-New Jersey-Maryland market in the United States (PJM), which contains hourly zonal prices in the Commonwealth Edison (COMED) and two day-ahead forecasts of load at the system and COMED zonal levels. The remaining three markets are obtained from the integrated European Power Exchange (EPEX). Belgium (EPEX-BE) and France (EPEX-FR) markets share the day-ahead forecast generation in France as covariates since it is known to be one of the best predictors for Belgian prices (Lago et al., 2018b). Finally, the German market (EPEX-DE) contains hourly prices, day-ahead load forecasts, and the country wind and solar generation day-ahead forecast.

Figure 4.4 displays the NP electricity price time series and its corresponding covariate variables to illustrate the datasets. The NP market is the least volatile among the considered markets, since most of its power comes from hydroelectric generation, renewable source volatility is negligible, and zero spikes are rare. The PJM market is transitioning from coal generation to natural gas and some renewable sources, zero spikes are rare, but the system exhibits higher volatility than NP. In EPEX-BE and EPEX-FR markets, negative prices and spikes are more frequent, and as time passes, these markets begin to show increasing signs of integration. Finally, the EPEX-DE market shows few price spikes, but the most frequent negative and zero price events, due in great part to the impact of renewable sources.

The exogenous covariates are normalized following best practices drawn from the EPF literature (Uniejewski et al., 2018). Preprocessing the inputs of neural networks is essential to accelerate and stabilize the optimization (LeCun et al., 1998).

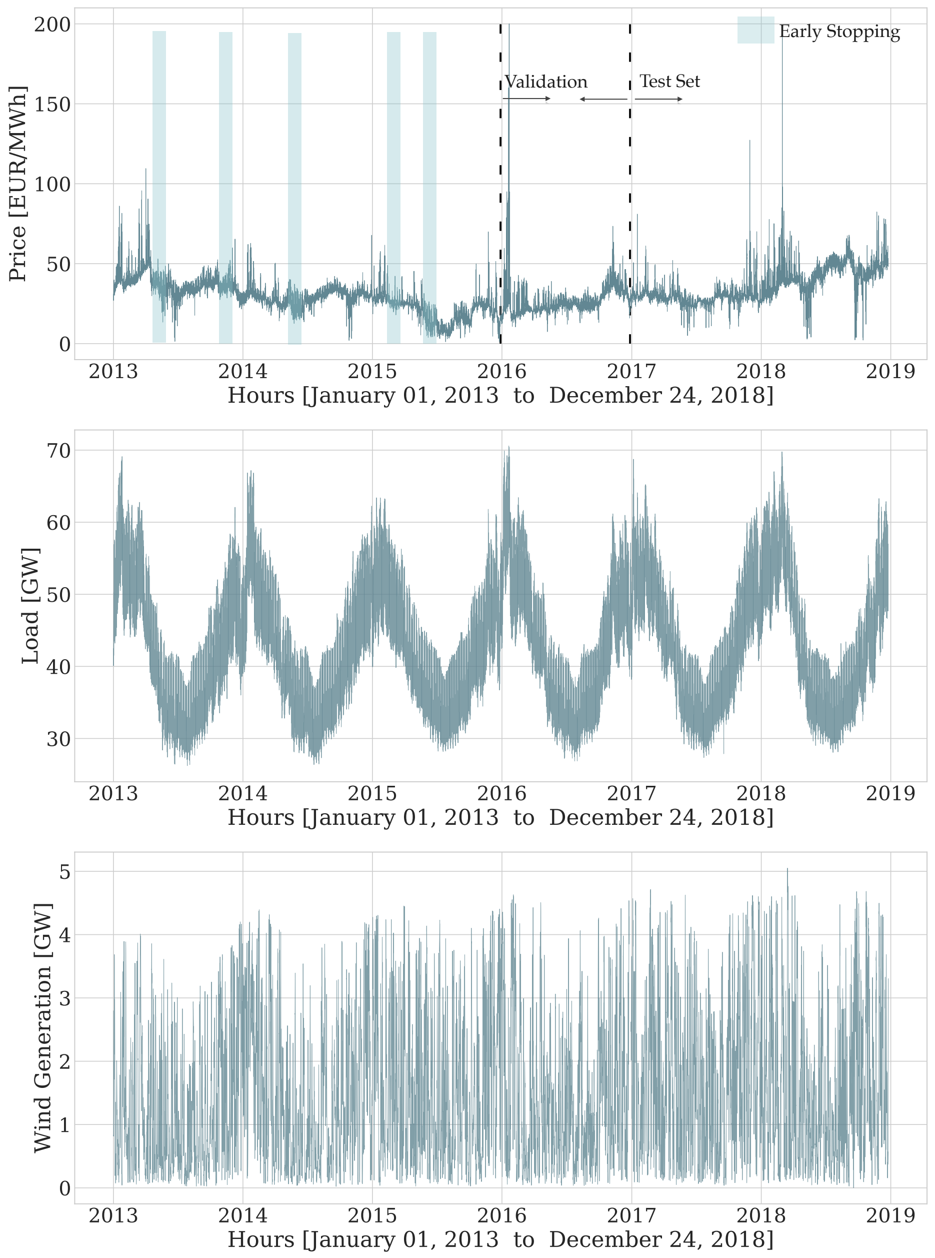


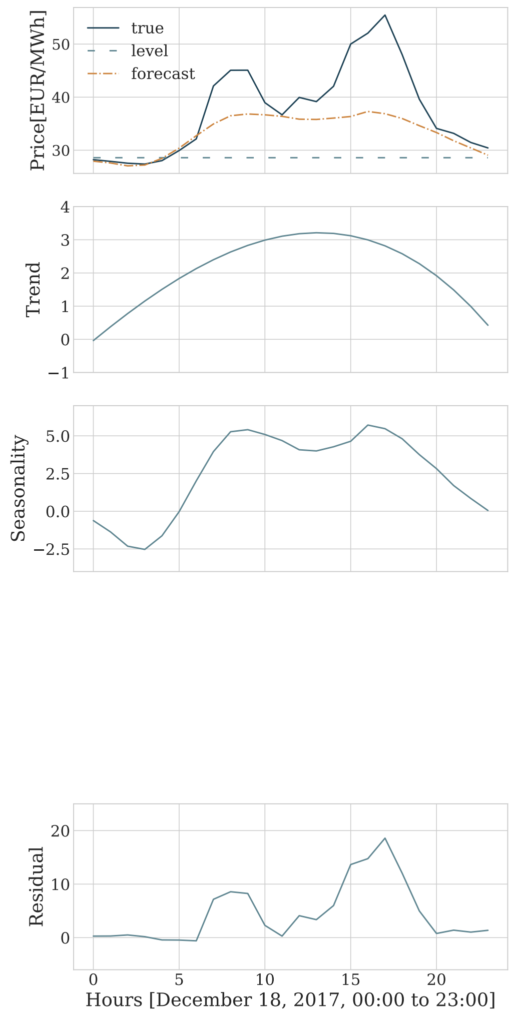
Figure 4.4: The top panel shows the *day-ahead* electricity price time series for the NordPool (NP) market. The second and third panels show the day-ahead forecast for the system load and wind generation. The training data is composed of the first four years of each dataset. The validation set is the year that follows the training data (between the first and second dotted lines). For the held-out test set, the last two years of each dataset are used (marked by the second dotted line). During *evaluation*, we recalibrate the model updating the training set to incorporate all available data before each daily prediction. The recalibration uses an early stopping set of 42 weeks randomly chosen from the updated training set (a sample selection is marked with blue rectangles in the top panel).

4.5.2 Interpretable Time Series Decomposition

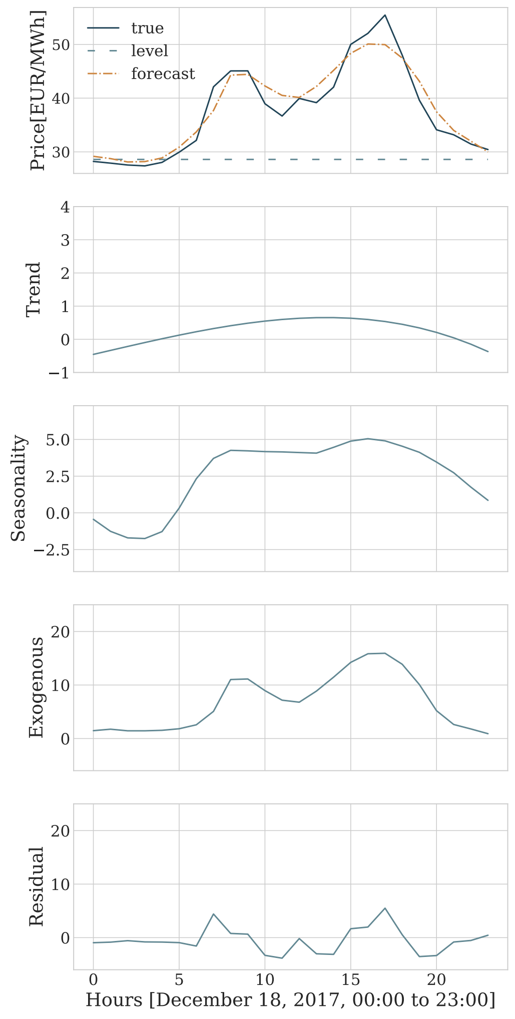
We demonstrate NBEATSx versatility and show how a careful selection of the inductive bias, constituted by the assumptions used to learn the modeled signal, endows it with an outstanding ability to model complex dynamics while enabling human understanding of its outputs.

Our method combines the power of non-linear transformations provided by neural networks with the flexibility to model multiple seasons that can be fractional, and simultaneously account for interaction events such as holidays and other covariates. As described earlier, the interpretable configuration of the NBEATSx architecture computes time-varying coefficients for slowly changing polynomial functions to model the trend, harmonic functions to model the cyclical behavior of the signal, and exogenous covariates. Here, we show how this configuration can decompose a time series into the classic set of level, trend, and seasonality components, while identifying the covariate effects.

In this time series signal decomposition example, we show how the NBEATSx-I model benefits over NBEATS-I from explicitly accounting for information carried by exogenous covariates. Figure 4.5 shows the NP electricity market's hourly price (EUR/MWh), for December 18, 2017 which is a day with high prices due to high load. Other days have a less pronounced difference between the results obtained with the original NBEATS-I and the NBEATSx-I. We selected a day with a higher than normal load for exposition purposes, to demonstrate qualitative differences in the forecasts. We can see a substantial difference in the forecast residual magnitudes in the bottom row of Figure 4.5. NBEATS shows a strong negative bias. On the other hand, NBEATSx-I is able to capture the evidently substantial explanatory value of the exogenous features, resulting in a much more accurate forecast.



(a) NBEATS



(b) NBEATSx

Figure 4.5: Time series signal decomposition for NP electricity price day-ahead forecasts using interpretable variants of NBEATS and NBEATSx. The top row of graphs shows the original signal and the level, the latter is defined as the last available observation before the forecast. The second row shows the polynomial trend components, the third and fourth rows display the complex seasonality modeled by nonlinear Fourier projections and the exogenous effects of the electricity load on the price, respectively. The bottom row graphs show the unexplained variation of the signal. The use of electricity load and production forecasts turns out to be fundamental for accurate price forecasting.

4.5.3 Evaluation

To ensure the comparability of our results with the existing literature, we opted to follow the widely accepted practice of evaluating the accuracy of point forecasts with the following metrics: Mean Absolute Error (MAE), relative Mean Absolute Error (relMAE) [1], symmetric Mean Absolute Percentage Error (sMAPE) and Root Mean Squared Error (RMSE), defined as:

$$\mathrm{MAE} = \frac{1}{24N_d}\sum_{d=1}^{N_d}\sum_{\tau=1}^{24}|y_{d,\tau}-\hat{y}_{d,\tau}| \qquad \mathrm{relMAE} = \frac{\sum_{d=1}^{N_d}\sum_{\tau=1}^{24}|y_{d,\tau}-\hat{y}_{d,\tau}|}{\sum_{d=1}^{N_d}\sum_{\tau=1}^{24}|y_{d,\tau}-\hat{y}_{d,\tau}^{Naive}|}$$

$$\mathrm{sMAPE} = \frac{200}{24N_d}\sum_{d=1}^{N_d}\sum_{\tau=1}^{24}\frac{|y_{d,\tau}-\hat{y}_{d,\tau}|}{|y_{d,\tau}|+|\hat{y}_{d,\tau}|} \qquad \mathrm{RMSE} = \sqrt{\frac{1}{24N_d}\sum_{d=1}^{N_d}\sum_{\tau=1}^{24}(y_{d,\tau}-\hat{y}_{d,\tau})^2}$$

where $\mathbf{y}_d = \mathbf{y}_{[t+1:t+H]}$ and $y_{d,\tau}$ and $\hat{y}_{d,\tau}$ are the actual value and the forecast of the time series at day d and hour τ, for our experiments given the two years of each test set $N_d = 728$. The MAE and RMSE measure the errors in absolute terms, and as such they are not easily comparable across time series. For this reason, we also include the sMAPE and relMAE metrics that are relative. The sMAPE is included as an alternative to MAPE which in the presence of values close to zero may degenerate (Hyndman and Koehler, 2006). The difference between the *relative mean absolute error* (relMAE) and *mean absolute scaled error* (MASE) is that the *relative mean absolute error* includes the out-of-sample predictions which helps to compare the relative performance of the model with the accuracy of the seasonal naive benchmark (Lago et al., 2021a).

Statistical Tests

To assess which forecasting model provides better predictions, we rely on the Giacomini-White test (GW) (Giacomini and White, 2006) of the multi-step conditional predictive ability, which can be interpreted as a generalization of the Diebold-Mariano test (DM) (Diebold and Mariano, 2002), widely used in the forecasting literature. Compared with the DM or other unconditional tests, the GW test is valid under general assumptions such as heterogeneity rather than stationarity of data. The GW test examines the null hypothesis of equal accuracy specified in Equation (4.12), measured by the $L1$ norm of the daily errors of a pair of models A and B, conditioned on the available information to that moment[2] in time $\mathcal{F}_{d-1}$.

$$H_0 : \mathbb{E}\left[||\mathbf{y}_d - \hat{\mathbf{y}}_d^A||_1 - ||\mathbf{y}_d - \hat{\mathbf{y}}_d^B||_1 \mid \mathcal{F}_{d-1}\right] \equiv \mathbb{E}\left[\Delta_d^{A,B} \mid \mathcal{F}_{d-1}\right] = 0 \tag{4.12}$$

[1]The Naive forecast method in EPF corresponds to a similar day rule, where the forecast for a Monday, Saturday and Sunday equals the value of the series observed on the same weekday of the previous week, while the forecast for Tuesday, Wednesday, Thursday, and Friday is the value observed on the previous day.

[2]In practice, the available information $\mathcal{F}_{d-1}$ is replaced with a constant and lags of the error difference $\Delta_d^{A,B}$ and the test is performed using a linear regression with a Wald-like test. When the conditional information considered is only the constant variable, one recovers the original DB test.

4.5.4 Train and Hyperparameter Optimization

Train Methodology

The cornerstone of the training methodology for NBEATSx and the benchmark models included in this work is the definition and use of the training, validation, early stopping, and test datasets depicted in Figure 4.4. The training set for the five markets comprises the first three years, the test set includes the last two years of data. The validation set is defined as the year between the train and test set coverages. The early stopping set, used for regularization, is either randomly sampled or corresponds to 42 weeks following the time span of the training set. These sets are used in the *hyperparameter optimization phase* and *recalibration phase* that we describe below.

During the *hyperparameter optimization phase*, model performance measured on the validation set is used to guide the exploration of the hyperparameter space defined in Table 4.2. During the *recalibration phase*, the optimally selected model, as defined by its hyperparameters, is re-trained for each day to include newly available information before the test inference. In this phase, an early stopping set provides a regularization signal for the retraining optimization.

To train the neural network, we use *mean absolute error* (MAE) Empirical Risk Minimization (ERM) optimized using Adaptive Moment Estimation SGD (ADAM) (Kingma and Ba, 2014)). Figure A.1 in the Appendix compares the training and validation trajectories for NBEATS and NBEATSx, as diagnostics to assess the differences of the methods. The early stopping strategy halts the training procedure if a specified number of consecutive iterations occur without improvements of the loss measured on the early stopping set (Yao et al., 2007).

The NBEATSx model is implemented and trained in PyTorch (Paszke et al., 2019) and can be run with both CPU and GPU resources. The code is available publicly in a dedicated repository at NBEATSx repository to promote reproducibility of the presented results and to support related research.

Hyperparameter Optimization

We follow the practice of Lago et al., 2018a to select the hyperparameters that define the model, input features, and optimization settings. During this phase, the validation dataset is used to guide the search for well performing configurations. To compare the benchmarks and the NBEATSx, we rely on the same automated selection process: a Bayesian optimization technique that efficiently explores the hyperparameter space using tree-structured Parzen estimators (HYPEROPT; Bergstra et al. 2011). The architecture, optimization, and regularization hyperparameters are summarized in Table 4.2. To have comparable results, during the hyperparameter optimization stage we used the same number of configurations as in Lago et al., 2018a.

Note, that some of the methods do not require any hyperparameter optimization – e.g., the AR1 benchmark – and some might only have one hyper-parameter to be determined, such as the regularization parameter in the LEARx method, which is typically computed using the information criteria or cross-validation.

Table 4.2: Hyperparameters of NBEATSx networks. They are common to all presented datasets. We list the typical values we considered in our experiments. The configuration that performed best on the validation set was selected automatically.

Hyperparameter	Considered Values
Architecture Parameters	
Input size, size of autorregresive feature window.	$L \in \{168\}$
Output size is the forecast horizon for day ahead forecasting.	$H \in \{24\}$
List for architecture's type/number of stacks.	$\{[\texttt{identity}, \texttt{TempConv}]\}$.
Type of activations used accross the network.	{softplus,selu,prelu,sigmoid}
Blocks separated by residual links per stack (shared across stacks).	{[1,1,1], [1, 1]}.
MLP layers within each block.	$\{2\}$
MLP hidden neurons on each layer of a block.	$N_h \in \{50, \ldots, 500\}$
Temporal convolution filters, for exogenous variables.	$N_f \in \{2, \ldots, 10\}$
Only interpretable, degree of trend polynomials.	$N_{pol} \in \{2\}$
Only interpretable, number of Fourier basis (seasonality smoothness).	$N_{har} \in \{6\}$
Whether NBEATSx coefficients take input $\mathbf{X}$	$\{\text{True}, \text{False}\}$
Optimization and Regularization parameters	
Initialization strategy for network weights.	{orthogonal, he_norm, glorot_norm}
Initial learning rate for regression problem.	Range(5e-4,1e-2)
The number of samples for each gradient step.	{256, 512}
The decay constant allows large initial lr to escape local minima.	{0.5}
Number of times the learning rate is halved during train.	{3}
Maximum number of iterations of gradient descent.	{30000}
Iterations without validation loss improvement before stop.	{10}
Frequency of validation loss measurements.	{100}
Whether batch normalization is applied after each activation.	{True, False}
The probability for dropout of neurons in the projection layers.	Range(0,1)
The probability for dropout of neurons for the exogenous encoder.	Range(0,1)
Constant to control the Lasso penalty used on the coefficients.	Range(0, 0.1)
Constant that controls the influence of L2 regularization of weights.	Range(1e-5,1e-0)
The objective loss function with which NBEATSx is trained.	{MAE}
Random weeks from full dataset used to validate.	{42}
Number of iterations of hyperparameter search.	{1500}
Random seed that controls initialization of weights.	DiscreteRange(1,1000)
Data Parameters	
Rolling window sample frequency, for data augmentation.	{1, 24}
Number of time windows included in the full dataset.	4 years
Number of validation weeks used for early stopping strategy.	{40, 52}
Normalization strategy of model inputs.	{none, median, invariant }

4.5.5 Ensembling

In many recent forecasting competitions, and particularly in the M4 competition, most of the top-performing models were ensembles (Atiya, 2020). It has been shown that in practice, combining a diverse group of models can be a powerful form of regularization to reduce the variance of predictions (Breiman, 1996; Nowotarski et al., 2014; Hubicka et al., 2018).

The techniques used by the forecasting community to induce diversity in the models are plentiful. The original NBEATS model obtained its diversity from three sources, training with different loss functions, varying the size of the input windows, and bagging models with different random initializations (Oreshkin et al., 2020). They used the median as the aggregation function for 180 different models. Interestingly, the original model did not rely on regularization, such as L2 or dropout, as Oreshkin et al., 2020 found it to be good for the individual models but detrimental to the ensemble.

In our case, we ensemble the NBEATSx model using two sources of diversity. The first being a data augmentation technique controlled by the sampling frequency of the windows used during training, as defined in the data parameters from Table 4.2. The second source of diversity being whether we randomly select the early stopping set or instead use the last 42 weeks preceding the test set. Combining the data augmentation and early stopping options, we obtain four models that we ensemble using arithmetic mean as the aggregation function. This technique is also used by the DNN benchmark (Lago et al., 2018a; Lago et al., 2021a).

4.5.6 Forecasting Results

We conducted an empirical study involving two types of Autorregresive Models (AR1 and ARx1) (Weron, 2014), the *Lasso Estimated Auto-Regressive* (LEARx) (Uniejewski et al., 2016), a parsimonious *Deep Neural Network* (DNN) (Lago et al., 2018a; Lago et al., 2021a), the original Neural Basis Expansion Analysis (NBEATS) without exogenous covariates (Oreshkin et al., 2020), and the Exponential Smoothing Recurrent Neural Network (ESRNN) (Smyl, 2019). This experiment examines the effects of including the covariate inputs and comparing NBEATSx with state-of-the-art methods for the electricity price day-ahead forecasting task.

Table 4.3 summarizes the performance of the ensembled models where NBEATSx ensemble shows prevailing performance. It improves 18.77% on average for all metrics and markets when compared with the original NBEATS and 20.6% when compared to ESRNN without time-dependent covariates. For the ensembled models, NBEATSx RMSE improved on average 4.68%, MAE improved 2.53%, relMAE improved 1.97%,and sMAPE improved 1.25%. When comparing NBEATSx ensemble against DNN ensemble on individual markets, NBEATSx improved by 5.38% on the NordPool market, by 2.48% on French market and 2.81% on German market. There was a non-significant difference of NBEATSx performance on PJM and EPEX-BE markets of 0.24% and 1.1%, respectively.

Figure 4.6 provides a graphical representation of the statistical significance from the *Giacomini-White* test (GW) for the six ensembled models, across the five markets for the MAE evaluation metric. A similar significance analysis is conducted for the single models. The models included

Table 4.3: Forecast accuracy measures for day-ahead electricity price predictions of *ensembled models*. The ESRNN and NBEATS do not include time dependent covariates. The reported metrics are *mean absolute error* (MAE), *relative mean absolute error* (rMAE), *symmetric mean absolute percentage error* (sMAPE) and *root mean squared error* (RMSE). The smallest errors in each row are highlighted in bold.
* The EPEX-DE's LEARxresults differ from Lago et al., 2021a – the values are revised (Lago et al., 2021b)

		AR1	ESRNN	NBEATS	ARx1	LEARx*	DNN	NBEATSx-G	NBEATSx-I
NP	MAE	2.26	2.09	2.08	2.01	1.74	1.68	**1.58**	1.62
	rMAE	0.71	0.66	0.66	0.63	0.55	0.53	**0.50**	0.51
	sMAPE	6.47	6.04	5.96	5.84	5.01	4.88	**4.63**	4.70
	RMSE	4.08	3.89	3.94	3.71	3.36	3.32	**3.16**	3.27
PJM	MAE	3.83	3.59	3.49	3.53	3.01	**2.86**	2.91	2.90
	rMAE	0.79	0.74	0.72	0.73	0.62	**0.59**	0.60	0.60
	sMAPE	14.5	14.12	13.57	13.64	11.98	**11.33**	11.54	11.61
	RMSE	6.24	5.83	5.64	5.74	5.13	5.04	5.02	**4.84**
EPEX-BE	MAE	7.2	6.96	6.84	7.19	6.14	**5.87**	5.95	6.11
	rMAE	0.88	0.85	0.83	0.88	0.75	**0.72**	0.73	0.75
	sMAPE	16.26	15.84	15.80	16.11	14.55	**13.45**	13.86	14.02
	RMSE	18.62	16.84	17.13	18.07	15.97	15.97	**15.76**	15.80
EPEX-FR	MAE	4.65	4.65	4.74	4.56	3.98	3.87	3.81	**3.79**
	rMAE	0.78	0.78	0.80	0.76	0.67	0.65	**0.64**	**0.64**
	sMAPE	13.03	13.22	13.30	12.7	11.57	10.81	**10.59**	10.69
	RMSE	13.89	11.83	12.01	12.94	**10.68**	11.87	11.50	11.25
EPEX-DE	MAE	5.74	5.60	5.31	4.36	3.61	3.41	3.31	**3.29**
	rMAE	0.71	0.70	0.66	0.54	0.45	0.42	**0.41**	**0.41**
	sMAPE	21.37	20.97	19.61	17.73	14.74	14.08	**13.99**	**13.99**
	RMSE	9.63	9.09	8.99	7.38	6.51	5.93	5.72	**5.65**

in the significance tests are the same as in Table 4.3: LEAR, DNN, ESRNN, NBEATS, and our proposed methods, NBEATSx-G and NBEATSx-I. The p-value of each comparison shows if the improvement of the model's predictions corresponding to the column index of a cell in the grids shown in Figure 4.6 over the model's predictions corresponding to the row of this cell of the grid is statistically significant. NBEATSx-G model outperformed DNN model in NP and EPEX-DE, while NBEATSx-I outperformed it in NP, EPEX-FR, and EPEX-DE. Moreover, no benchmark significantly outperformed NBEATSx-I and NBEATSx-G in any market.

In the Appendix A we observe similar results for the single best models chosen from the four possible configurations of the ensemble components described in Section 4.5.5. Table A.2 summarizes the accuracy of the predictions measured with the MAE and Figure A.2 displays the significance of the GW test. Ensembling improves the accuracy of NBEATSx by 3% on average acrosss all markets, when compared to the single best models.

Finally, regarding the computational time complexity NBEATSx maintains good performance. As shown in Table A.1 in the Appendix, the time necessary to compute day-ahead predictions is in the order of miliseconds and comparable to that of the LEAR and DNN benchmarks. Additionally, the average time needed to perform a recalibration only takes circa 50 percent more than the relatively parsimonious DNN.

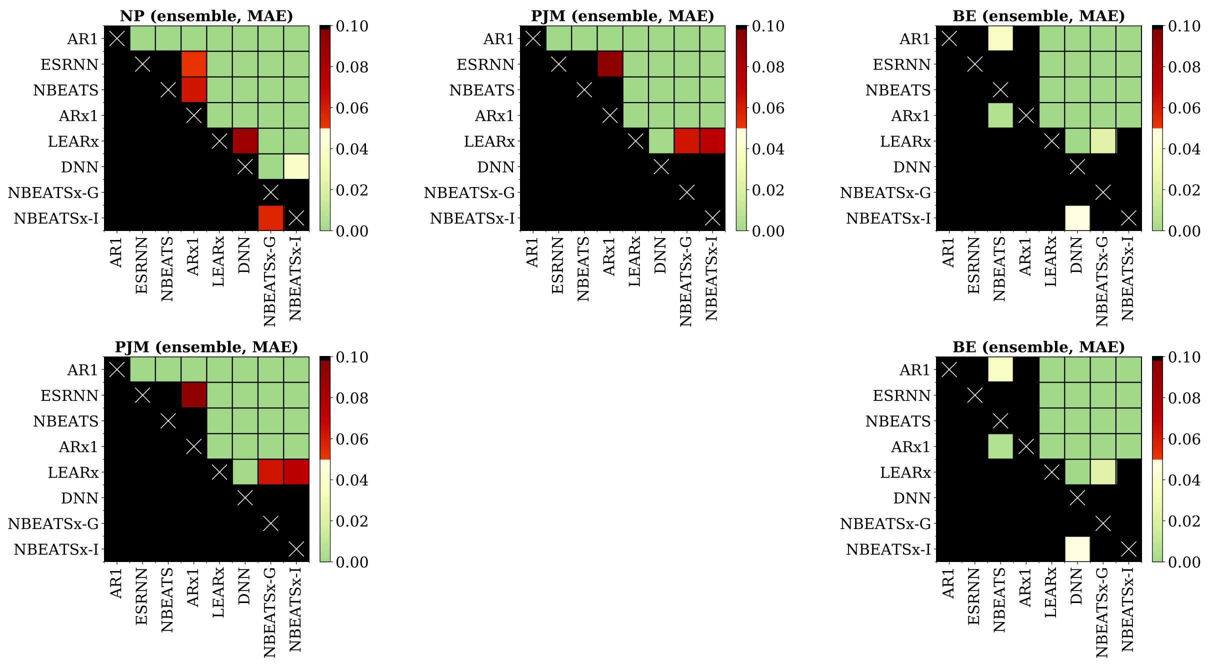


Figure 4.6: Giacomini-White test for the day-ahead predictions with *mean absolute error* (MAE) applied to pairs of the ensembled models on the five electricity markets datasets. Each grid represents one market. Each colored cell in a grid is plotted black, unless the predictions of the model corresponding to its column of the grid outperforms the predictions of the model corresponding to its row of the grid. The color scale reflects significance of the difference in MAE, with solid green representing the lowest p-values.

4.6 Conclusion

We have presented NBEATSx: the new method for univariate time series forecasting with exogenous variables. It extends the well-performing neural basis expansion analysis. The resulting neural based method has several valuable properties that make it suitable for a wide range of forecasting tasks. The network is fast to optimize as it is mainly composed of fully-connected layers. It can produce interpretable results, and achieves state-of-the-art performance on forecasting tasks where consideration of exogenous variables is fundamental.

We demonstrated NBEATSx's utility using a set of benchmark datasets from electricity price forecasting domain, but it can be straightforwardly applied to forecasting problems in other domains. Qualitative evaluation shows that the interpretable configuration of NBEATSx can provide valuable insights to the analyst, as it explains the variation of the time series by separating it into trend, seasonality, and exogenous components, in a fashion analogous to classic time series decomposition. Regarding the quantitative forecasting performance, we observed no significant differences between ESRNN and NBEATS without exogenous variables. At the same time, NBEATSx improves over NBEATS by nearly 20% and up to 5% over LEAR and DNN models specialized for the Electricity Price Forecasting tasks. Finally, we found no significant trade-offs between the accuracy and interpretability of NBEATSx-G and NBEATSx-I predictions.

The neural basis expansion analysis is a flexible method capable of producing accurate and interpretable forecasts, yet there is still room for improvement. For instance, augmentation of the harmonic functions towards wavelets or replacement of the convolutional encoder that would generate the covariate basis with smoothing alternatives such as splines. Additionally, one can extend the non-interpretable method by regularizing its outputs with smoothness constraints.

Chapter 5

Probabilistic Hierarchical Forecasting

5.1 Summary

In this case study, we address the hierarchical forecasting challenge that arises when time series data is organized into natural groups with multiple levels of aggregation, for which we need accurate predictions that maintain probabilistic coherence. Motivated by the shortcomings of existing methods, which often lack accuracy or are computationally complex, we propose a novel approach that combines the strengths of neural networks with a novel probabilistic mixture model. Our composite method is accurate, computationally efficient, and probabilistically coherent by construction.

5.2 Motivation

Large collections of time series data are commonly organized into structures with different levels of aggregation; examples include product and geographical groupings. It is often important to ensure that the forecasts are coherent so that the predicted values at disaggregate levels add up to the aggregate forecast (Hyndman et al., 2014; Athanasopoulos et al., 2017; Spiliotis et al., 2020b). Hierarchical coherency is easy to understand for mean forecasts which are additive, for probabilistic forecast the notion extends and probabilistic coherency is achieved when the forecast distribution of the aggregate series is identical to the distribution of the sum of its children's forecast series under an implicit or explicit joint distribution (Ben Taieb et al., 2017; Ben Taieb et al., 2021; Wickramasuriya, 2023; Panagiotelis et al., 2020; Panagiotelis et al., 2023).

While summing the most disaggregated level forecasts (*bottom-up*) will provide coherent forecasts, it can perform poorly on highly disaggregated series. Existing solutions rely on reconciliation methods that first generate independent *base* forecasts for each series and then reconcile them to produce coherent forecasts (Hyndman et al., 2011; Wickramasuriya et al., 2019). Efficiently leveraging the *cross-learning* approach (Makridakis et al., 2018c; Semenoglou et al., 2021) to improve accuracy while maintaining probabilistic coherence remains a challenge.

This chapter presents a novel method for producing probabilistic coherent forecasts. It combines the strength of modern neural networks and an intuitive statistical model for the disaggregated-forecast joint distribution. In contrast to earlier efforts (Han et al., 2021; Rangapuram et al., 2021), our approach is an extension to the *Mixture Density Network* (MDN; Bishop 1994). Our method is coherent by construction and does not require an explicit reconciliation step, either as part of a single end-to-end network or as a separate step. We call it the *Hierarchical Forecast Network* (HINT). The HINT models the joint probability of the multivariate time series as a finite mixture of Poisson distributions and combines that with the well-established MQForecaster neural architecture (Wen et al., 2017; Eisenach et al., 2021). We formulate the problem as an MDN, where we independently choose a relevant class of probabilistic model and the neural architecture. Our method's key advantages are:

(i) **Flexible Forecast Distribution**: We model the predictive distribution as a finite mixture of Poisson random variables, which is analogous to a Poisson kernel density. The resulting distribution is flexible, capable of accurately modeling a wide range of joint probability distributions, compatible as an output layer with state-of-the-art neural architectures. We will demonstrate this empirically on three different forecasting tasks in Section 5.5.

(ii) **Computational Efficiency**: Learning coherent forecast distributions in a high dimensional hierarchical space can be computationally intractable. To alleviate this, we anchor the HINT on a bottom level multivariate distribution and employ composite likelihood optimization strategies, which enables us to extend to large-scale applications.

(iii) **Hierarchical Forecasting Comparison**: We demonstrate the effectiveness of the HINT model on three hierarchical forecasting tasks achieving state-of-the-art results on two. Our approach improves probabilistic coherent accuracy by 11.8% on Australian domestic tourism data and 8.1% on the Favorita grocery sales dataset. It performs similarly to statistical baselines on a San Francisco Bay Area dataset.

5.3 Related Work

5.3.1 Hierarchical Forecasting Notation

Mathematically a hierarchical series can be denoted by the vector $\mathbf{y}_{[a,b],\tau} = [\,\mathbf{y}^{\top}_{[a],\tau} \mid \mathbf{y}^{\top}_{[b],\tau}\,]^{\top} \in \mathbb{R}^{(N_a+N_b)}$ for each time point τ; where $[a]$, $[b]$ stand for the set of all aggregate and bottom indices of the time series respectively, and $\alpha \in [a], \beta \in [b]$ are single aggregate and bottom time series indices. The total number of series in the hierarchy is $|\,[a,b]\,| = (N_a + N_b)$, where $|\,[a]\,| = N_a$ is the number of aggregated series and $|\,[b]\,| = N_b$ the number of bottom series that are at the most dis-aggregated level possible. Time indices for past information until t is given by the set $[t]$ with length $|\,[t]\,| = N_t$, and the h time step forecast horizon is denoted as $[t+1:t+h]$. e use t to denote the forecast creation time, while τ identifies the relative forecast horizon index. With this notation the hierarchical aggregation constraints at each time point $\tau \in [t+1:t+h]$

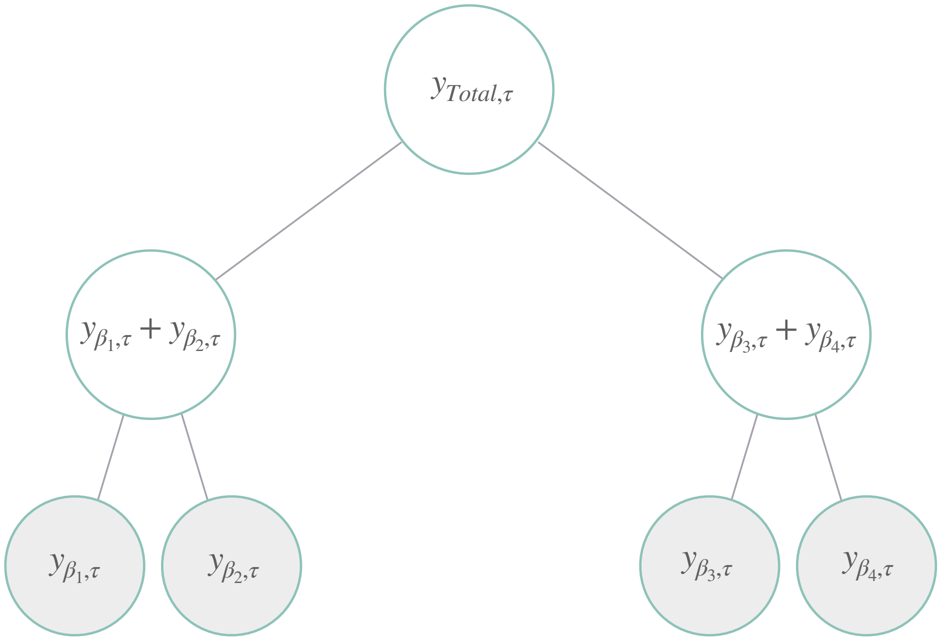


Figure 5.1: *A simple three level time series hierarchical structure, with four bottom level variables. The disaggregated bottom variables are marked with gray background. In this description each node represents non overlapping series for a single point in time.*

have the following matrix representation:

$$\mathbf{y}_{[a,b],\tau} = \mathbf{S}_{[a,b][b]}\mathbf{y}_{[b],\tau} \quad \Leftrightarrow \quad \begin{bmatrix}\mathbf{y}_{[a],\tau}\\ \mathbf{y}_{[b],\tau}\end{bmatrix} = \begin{bmatrix}\mathbf{A}_{[a][b]}\\ \mathbf{I}_{[b][b]}\end{bmatrix}\mathbf{y}_{[b],\tau} \tag{5.1}$$

The matrix $\mathbf{S}_{[a,b][b]} \in \mathbb{R}^{(N_a+N_b)\times N_b}$ aggregates the bottom level to the series above. It is composed by the aggregation matrix $\mathbf{A}_{[\mathrm{a}][\mathrm{b}]} \in \mathbb{R}^{N_a \times N_b}$ and an $N_b \times N_b$ identity matrix $\mathbf{I}_{[b][b]}$.

Figure 5.1 represents a hierarchy where each parent node is the sum of its children. Here the dimensions are $N_a = 3$, $N_b = 4$, and the hierarchical, aggregated and base series are respectively:

$$\begin{gathered} y_{\text{Total},\tau} = y_{\beta_1,\tau} + y_{\beta_2,\tau} + y_{\beta_3,\tau} + y_{\beta_4,\tau} \\ \mathbf{y}_{[a],\tau} = [y_{\text{Total},\tau},\ y_{\beta_1,\tau} + y_{\beta_2,\tau},\ y_{\beta_3,\tau} + y_{\beta_4,\tau}]^\intercal \qquad \mathbf{y}_{[b],\tau} = [y_{\beta_1,\tau},\ y_{\beta_2,\tau},\ y_{\beta_3,\tau},\ y_{\beta_4,\tau}]^\intercal \end{gathered} \tag{5.2}$$

The constraint matrix of the Figure 5.1 example and the corresponding aggregations from Equation (5.2) is the following:

$$\mathbf{S}_{[a,b][b]} = \begin{bmatrix} \mathbf{A}_{[\mathrm{a}][\mathrm{b}]} \\ \\ \mathbf{I}_{[\mathrm{b}][\mathrm{b}]} \end{bmatrix} = \left[\begin{array}{cccc} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ \hline 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right]$$

5.3.2 Mean Forecast Reconciliation Strategies

Statistical solutions for hierarchical forecasting implement a two-stage process, first generating base forecasts $\hat{\mathbf{y}}_{[a,b],\tau} \in \mathbb{R}^{N_a+N_b}$ and then reconciling them into coherent forecasts $\tilde{\mathbf{y}}_{[a,b],\tau}$.

Definition 5.1: Hierarchical Reconciliation

Hierarchical reconciliation can be compactly expressed by:

$$\tilde{\mathbf{y}}_{[a,b],\tau} = \mathbf{S}_{[a,b][b]}\mathbf{P}_{[b][a,b]}\hat{\mathbf{y}}_{[a,b],\tau} \tag{5.3}$$

where $\mathbf{S}_{[a,b][b]} \in \mathbb{R}^{(N_a+N_b)\times N_b}$ is the hierarchical aggregation matrix, $\mathbf{P}_{[b][a,b]} \in \mathbb{R}^{N_b\times(N_a+N_b)}$ is a matrix determined by the reconciliation strategies.

The most common reconciliation methods can be classified into *top-down*, *bottom-up* and *alternative* approaches[1].

- Bottom-up: The simple *bottom-up* strategy, abbreviated as NaiveBU (Orcutt et al., 1968), first generates bottom level forecasts and then aggregates them to produce forecasts for all the series in the multivariate structure. Here the reconciliation matrix is given by:

$$\mathbf{P}_{[b][a,b]} = [\mathbf{0}_{\mathrm{[b][a]}} \mid \mathbf{I}_{\mathrm{[b][b]}}] \tag{5.4}$$

- Top-down: The *top-down* strategy, abbreviated as TD (Gross and Sohl, 1990; Fliedner, 1999), disaggregates down the total forecast through hierarchy using proportions that can be historical actuals or forecasted separately. Its reconciliation matrix is given by:

$$\mathbf{P}_{[b][a,b]} = [\mathbf{p}_{\mathrm{[b]}} \mid \mathbf{0}_{\mathrm{[b][a,b}\ -1]}] \tag{5.5}$$

- Alternative: The more recent *middle-out* strategies, denoted as MO (Hyndman et al., 2011; Hyndman and Athanasopoulos, 2018a)), treat the second stage reconciliation as an optimization problem for the matrix $\mathbf{P}_{[b][a,b]}$. These reconciliation techniques include among others the *minimum trace* reconciliation (MinT; Wickramasuriya et al. 2019) and the *empirical risk minimization* approach (ERM; Souhaib and Bonsoo 2019).

Despite the advancements in alternative reconciliation strategies with statistical solutions, as mentioned in Section 5.2, there are still fundamental limitations. First, most post-process reconciliation methods produce mean forecasts but not probabilistic forecasts, with some exceptions that have relied on univariate statistical methods with strong probability assumptions for the base series that may be restrictive (Ben Taieb et al., 2017; Panagiotelis et al., 2020). Second, the mentioned methods independently learn the model parameters of the base level forecasts, limiting the base model's optimization inputs to single series. This approach induces an over-fitting prone setting for complex nonlinear methods, which, as noted by the forecasting community, is one of their biggest challenges (Makridakis et al., 2018b). The implied data scarcity translates into a missed opportunity to leverage the flexibility of nonlinear methods.

[1]For our work's purposes we implemented in Python a comprehensive collection of hierarchical reconciliation methods, we made them available in the HierarchicalForecast library (Olivares et al., 2022c).

5.3.3 Probabilistic Coherent Forecasting

There are few specialized methods on coherent probabilistic forecasting as most research on hierarchical forecasting has been limited to point predictions. Exceptions are the work by Shang and Hyndman, 2017 and Jeon et al., 2019 that provide an early exploration of forecast quantile reconciliation; Ben Taieb et al., 2017 and Ben Taieb et al., 2021 that propose the combination of bottom-level forecast marginal distributions with empirical copula functions describing their dependencies to create aggregate predictive distributions.

To the best of our knowledge, a formal definition of probabilistic coherence has only been explored by Ben Taieb et al. 2021, Puwasala et al. 2018, Wickramasuriya 2023 and Panagiotelis et al. 2023. Ben Taieb et al. 2021 provides a convolution-based definition while Panagiotelis et al. 2023 provide a generalized and intuitive framework[2] that we follow in our work and introduce:

Definition 5.2: Probabilistic Coherence

Let $(\Omega_{[b]}, \mathcal{F}_{[b]}, \hat{\mathbb{P}}_{[b]})$ be a probabilistic forecast space, with sample space $\Omega_{[b]}$, $\mathcal{F}_{[b]}$ its σ-algebra, and $\hat{\mathbb{P}}_{[b]}$ a forecast probability. Let $\mathbf{S}_{[a,b][b]}(\cdot) : \Omega_{[b]} \mapsto \Omega_{[a,b]}$ be the linear transformation implied by the constraints matrix. A coherent probabilistic forecast space $(\Omega_{[a,b]}, \mathcal{F}_{[a,b]}, \hat{\mathbb{P}}_{[a,b]})$ satisfies:

$$\hat{\mathbb{P}}_{[a,b]}\left(\mathbf{S}_{[a,b][b]}(\mathcal{B})\right) = \hat{\mathbb{P}}_{[b]}\left(\mathcal{B}\right) \quad \text{for any set } \mathcal{B} \in \mathcal{F}_{[b]} \text{ and set's image } \mathbf{S}_{[a,b][b]}(\mathcal{B}) \in \mathcal{F}_{[a,b]} \tag{5.6}$$

i.e., it assigns a zero probability to sets in $\mathbb{R}^{N_a+N_b}$ not containing any coherent forecasts.

For a simple definition-satisfying example, consider three random variables $(Y_\alpha, Y_{\beta_1}, Y_{\beta_2})$ with $Y_\alpha := Y_{\beta_1} + Y_{\beta_2}$. A coherent forecast assigns zero probability to the variable realizations $(y_\alpha,\ y_{\beta_1},\ y_{\beta_2})$ if they do not satisfy the aggregation constraint $y_\alpha = y_{\beta_1} + y_{\beta_2}$. An equivalent definition of coherence is to require that the marginal distributions are derivable from the joint distribution of the bottom level random variables. In this case, the probability function of interest is $\hat{\mathbb{P}}(Y_{\beta_1}, Y_{\beta_2})$, and the marginal probabilities can be derived from it using indicator functions as follows:

$$\hat{\mathbb{P}}(Y_{\beta_1} = y_{\beta_1}) = \sum_{y_{\beta_2}} \hat{\mathbb{P}}(Y_{\beta_1} = y_{\beta_1}, Y_{\beta_2} = y_{\beta_2}) \quad \text{and} \quad \hat{\mathbb{P}}(Y_{\beta_2} = y_{\beta_2}) = \sum_{y_{\beta_1}} \hat{\mathbb{P}}(Y_{\beta_1} = y_{\beta_1}, Y_{\beta_2} = y_{\beta_2})$$

$$\hat{\mathbb{P}}(Y_\alpha = y_\alpha) = \sum_{y_{\beta_1}, y_{\beta_2}} \hat{\mathbb{P}}(Y_{\beta_1} = y_{\beta_1}, Y_{\beta_2} = y_{\beta_2})\, \mathbb{1}(y_\alpha = y_{\beta_1} + y_{\beta_2})$$

where $\mathbb{1}(y_\alpha = y_{\beta_1} + y_{\beta_2})$ equals 1 when $y_\alpha = y_{\beta_1} + y_{\beta_2}$, and 0 otherwise. The marginal probability for Y_α, has the aggregation constraint built into its definition and is by construction a hierarchically coherent probability with respect to $\hat{\mathbb{P}}(Y_{\beta_1})$ and $\hat{\mathbb{P}}(Y_{\beta_2})$. Note that knowledge of the joint distribution is not required to generate hierarchically coherent forecast. However, if we had access to the joint distribution constructing hierarchically coherent marginal distributions becomes straight forward.

[2]Rangapuram et al., 2021 informally introduce a probabilistic coherence notion, with the `HierE2E` method.

5.3.4 Hierarchical Neural Forecasting

In the last decade, neural network-based forecasting methods have become ubiquitous in large-scale forecasting applications (Wen et al., 2017; Böse et al., 2017; Madeka et al., 2018; Eisenach et al., 2021), transcending industry boundaries into academia, as it has redefined the state-of-the-art in many practical tasks (Yu et al., 2018; Ravuri et al., 2021; Olivares et al., 2022a) and forecasting competitions (Makridakis et al., 2018c; Makridakis et al., 2020b).

The latest neural network-based solutions to the hierarchical forecasting include methods like the *Simultaneous Hierarchically Reconciled Quantile Regression* (SHARQ; Han et al. 2021) and *Hierarchically Regularized Deep Forecasting* (HIRED; Paria et al. 2021) and the *Probabilistic Robust Hierarchical Network* (PROFHiT; Kamarthi et al. 2022) that augment the training loss function with approximations to the hierarchical constraints. And *Hierarchical End-to-End* learning (HierE2E; Rangapuram et al. 2021) that integrates an alternative reconciliation strategy in its VAR forecasts through linear projections. With the exception of HierE2E, the rest of these methods encourage probabilistic coherence through regularization but do not guarantee it. Additionally, if a user requires updating the hierarchical structure of interest, a whole new optimization of the networks would be needed for the existing methods to forecast the structure correctly.

Our proposed method, HINT, addresses these deficiencies by specifying any network's output as our proposed Poisson Mixture. With this predictive distribution, a model needs only to forecast the hierarchy's bottom-most level, after which any desired hierarchical structure of interest can be predicted with guaranteed probabilistic hierarchical coherence.

5.4 Methodology

5.4.1 Multivariate Poisson Mixture Distribution

In this work, we focus our attention on hierarchical forecasting of non-negative discrete events. Many forecasting problems fall in this category as shown by the data sets described in Section 5.5. However, the general framework presented here works for continuous distributions as well with a suitably chosen base class for the mixture distribution e.g. Gaussian distribution. For discrete events, we start by postulating that the forecast joint probability of bottom level future multivariate time series realization $\mathbf{y}_{[b][t+1:t+h]} \in \mathbb{N}^{N_b \times h}$ is captured by the following mixture:

$$\begin{aligned}\hat{\mathbb{P}}\left(\mathbf{y}_{[b][t+1:t+h]} \middle|\, \boldsymbol{\lambda}_{[b][k][t+1:t+h]}\right) &= \sum_{\kappa=1}^{N_k} w_\kappa \prod_{(\beta,\tau)\in[b][t+1:t+h]} \mathrm{Poisson}\left(y_{\beta,\tau} | \lambda_{\beta,\kappa,\tau}\right) \\ &= \sum_{\kappa=1}^{N_k} w_\kappa \prod_{(\beta,\tau)\in[b][t+1:t+h]} \frac{\lambda_{\beta,\kappa,\tau}^{y_{\beta,\tau}}}{y_{\beta,\tau}!} e^{-\lambda_{\beta,\kappa,\tau}}\end{aligned} \tag{5.7}$$

The joint distribution in Equation (5.7), assumes that the modeled observations $\mathbf{y}_{[b][t+1:t+h]}$ are *conditionally independent* given the latent Poisson rates $\boldsymbol{\lambda}_{[b][k][t+1:t+h]}$. That is for all bottom

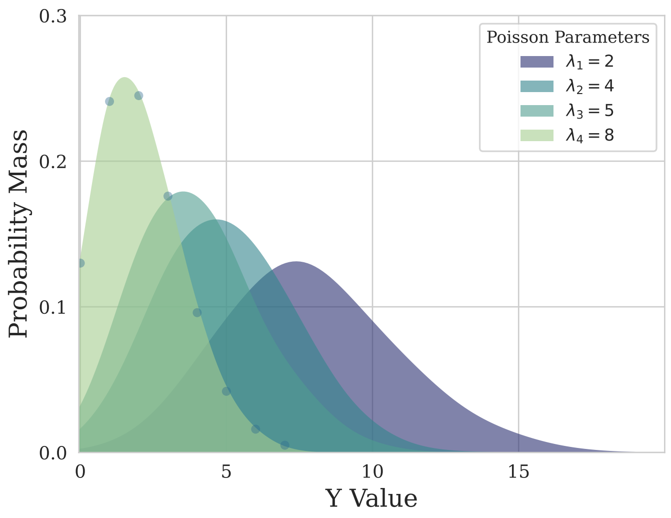


Figure 5.2: The Poisson Mixture distribution has desirable properties that make it well-suited for probabilistic hierarchical forecasting for count data. Under minimal conditions, its aggregation rule implies probabilistic coherence of the random variables it models.

level series and horizons $(\beta, \tau) \neq (\beta', \tau')$ and $(\beta, \tau), (\beta', \tau') \in [b][t+1:t+h]$:

$$\hat{\mathbb{P}}(Y_{\beta,\tau}, Y_{\beta',\tau'} | \lambda_{\beta,\kappa,\tau}, \lambda_{\beta',\kappa,\tau'}) = \hat{\mathbb{P}}(Y_{\beta,\tau} | \lambda_{\beta,\kappa,\tau}) \hat{\mathbb{P}}(Y_{\beta',\tau'} | \lambda_{\beta',\kappa,\tau'}) \tag{5.8}$$

The mixture model is able to describe single bottom level and their correlations through the distribution of the mixture's latent variables defined by the Poisson rates $\boldsymbol{\lambda}_{[b][k][t+1:t+h]} \in \mathbb{R}^{N_b \times N_k \times h}$ and the associated weights $\mathbf{w}_{[k]} \in [0,1]^{N_k}$, with $\mathbf{w}_{[k]} \geq 0$ and $\sum_{\kappa=1}^{N_k} w_\kappa = 1$. The number of components $\mid [k] \mid = N_k$ is a hyperparameter of the model that controls the flexibility of the mixture distribution. We show an example of the Poisson mixture distribution in Figure 5.2.

5.4.2 Marginal Distributions for Bottom Series

Equation (5.7) describes the joint distribution of all bottom level time series. We can derive the marginal distribution for one of the bottom level series $\beta \in [b]$ and for a single future time period $\tau \in [t+1:t+h]$ by integrating out the remaining time and series indices. The marginal distribution is:

$$\begin{aligned}\hat{\mathbb{P}}(Y_{\beta,\tau} = y_{\beta,\tau}) &= \sum_{\mathbf{y}_{[b][t+1:t+h] \setminus (\beta,\tau)}} \sum_{\kappa=1}^{N_k} w_\kappa \prod_{(\beta',\tau') \in [b][t+1:t+h]} \text{Poisson}(y_{\beta',\tau'} | \lambda_{\beta',\kappa,\tau'}) \\ &= \sum_{\kappa=1}^{N_k} w_\kappa \ \text{Poisson}(y_{\beta,\tau} | \lambda_{\beta,\kappa,\tau}) \end{aligned} \tag{5.9}$$

We get a clean final expression for the marginal distribution, which is equivalent to simply dropping all other time series and time periods from the product in Equation (5.7).

5.4.3 Marginal Distributions for Aggregate Series

An important consequence of the *conditional independence* in Equation (5.8) is that the marginal distributions at aggregate levels $\mathbf{y}_{[a],\tau}$ can be computed via simple component-wise addition of the lower level Poisson rates. For example consider an aggregate level variable $Y_{\alpha,\tau} = Y_{\beta_1,\tau} + Y_{\beta_2,\tau}$. The marginal distribution for $Y_{\alpha,\tau}$ can be derived from the joint distribution in Equation (5.7) as follows. First marginalize all other series and time indices (as done in Section 5.4.2 above), which gives us the joint distribution for $Y_{\beta_1,\tau}$ and $Y_{\beta_2,\tau}$

$$\hat{\mathbb{P}}(Y_{\beta_1,\tau} = y_{\beta_1,\tau}, Y_{\beta_2,\tau} = y_{\beta_2,\tau}) = \sum_{\kappa=1}^{N_k} w_\kappa \, \text{Poisson}(y_{\beta_1,\tau}|\lambda_{\beta_1,\kappa,\tau}) \times \text{Poisson}(y_{\beta_2,\tau}|\lambda_{\beta_2,\kappa,\tau})$$

Now, the aggregate marginal distribution is

$$\hat{\mathbb{P}}(Y_{\alpha,\tau} = y_{\alpha,\tau}) \quad = \quad \sum_{y_{\beta_1,\tau}, y_{\beta_2,\tau}} \hat{\mathbb{P}}(Y_{\beta_1,\tau} = y_{\beta_1,\tau}, Y_{\beta_2,\tau} = y_{\beta_2,\tau}) \, \mathbb{1}(y_{a,\tau} = y_{\beta_1,\tau} + y_{\beta_2,\tau})$$

For each mixture component κ, the distributions of $y_{\beta_1,\tau}$ and $y_{\beta_2,\tau}$ conditioned on respective Poisson rates $\lambda_{\beta_1,\kappa,\tau}$ and $\lambda_{\beta_2,\kappa,\tau}$ are independent Poisson distributions, and therefore, the distribution of the aggregate is another Poisson Mixture distribution with parameters $\lambda_{a,\kappa,\tau} = \lambda_{\beta_1,\kappa,\tau} + \lambda_{\beta_2,\kappa,\tau}$.

$$\hat{\mathbb{P}}(Y_{\alpha,\tau} = y_{\alpha,\tau}) = \sum_{\kappa=1}^{N_k} w_\kappa \text{Poisson}(y_{\alpha,\tau}|\lambda_{\alpha,\kappa,\tau} = \lambda_{\beta_1,\kappa,\tau} + \lambda_{\beta_2,\kappa,\tau})$$

The *aggregation rule* can be concisely described as:

$$\boldsymbol{\lambda}_{[a][k],\tau} = \mathbf{A}_{[\mathrm{a}][\mathrm{b}]} \boldsymbol{\lambda}_{[b][k],\tau} \tag{5.10}$$

with $\mathbf{A}_{[\mathrm{a}][\mathrm{b}]}$ the hierarchical aggregation matrix defined in Section 5.3.1. The joint predictive distribution is probabilistic coherent by construction. We offer a formal proof of HINT' satisfaction of the probabilistic coherence property from Definition 5.2 in Section 5.3.3.

5.4.4 Covariance Matrix

Using the law of total covariance and the *conditional independence* from Equation (5.8), we show in Appendix B.2 that the covariance of any two bottom level series naturally follows:

$$\text{Cov}(Y_{\beta,\tau}, Y_{\beta',\tau'}) = \overline{\lambda}_{\beta,\tau} \mathbb{1}(\beta = \beta')\mathbb{1}(\tau = \tau') + \sum_{\kappa=1}^{N_k} w_\kappa \left(\lambda_{\beta,\kappa,\tau} - \overline{\lambda}_{\beta,\tau}\right) \left(\lambda_{\beta',\kappa,\tau'} - \overline{\lambda}_{\beta',\tau'}\right) \tag{5.11}$$

where $\overline{\lambda}_{\beta,\tau} = \sum_{\kappa=1}^{N_k} w_\kappa \lambda_{\beta,\kappa,\tau}$. Appendix B.3 shows the non-diagonal covariance matrix expressivity, as determined by its rank, depends on the number of mixture components from Equation (5.7).

5.4.5 Parameter Estimation and Inference

Maximum Joint Likelihood

To estimate model parameters, we can use Maximum Likelihood Estimation (MLE) implied by the joint distribution from Equation (5.7). Let θ represent the neural network parameters as described in Section 5.4.6, we parameterize the probabilistic model with Poisson rates $\boldsymbol{\lambda}_{[b][k][t+1:t+h]}$ as follows:

$$\begin{aligned}\boldsymbol{\lambda}_{[b][k][t+1:t+h]} &:= \hat{\boldsymbol{\lambda}}_{[b][k][t+1:t+h]}(\theta \,|\mathbf{y}_{[b][:t]}, \mathbf{x}^{(h)}_{[b][:t]}, \mathbf{x}^{(f)}_{[b][t+1:t+h]}, \mathbf{x}^{(s)}_{[b]}) \\ \mathbf{w}_{[k]} &:= \hat{\mathbf{w}}_{[k]}(\theta \,|\tilde{\mathbf{x}}^{(h)}_{[:t]}, \tilde{\mathbf{x}}^{(s)})\end{aligned} \tag{5.12}$$

We condition the probability on the history of the bottom level time series, other associated historical covariates, future information available at the forecast generation time and static features, denoted by $\mathbf{y}_{[b][:t]}$, $\mathbf{x}^{(h)}_{[b][:t]}$, $\mathbf{x}^{(f)}_{[b][t+1:t+h]}$ and $\mathbf{x}^{(s)}_{[b]}$ respectively. And the shared mixture weights $\mathbf{w}_{[k]}$, are conditioned on temporal and static aggregate features shared across the bottom series, $\tilde{\mathbf{x}}^{(h)}_{[:t]}$ and $\tilde{\mathbf{x}}^{(s)}$ respectively.

We denote the combined conditioning information as

$$\mathbf{x}^{(h)}_{[:t]} = \{\mathbf{x}^{(h)}_{[b][:t]}, \tilde{\mathbf{x}}^{(h)}_{[:t]}\} \quad \text{and} \quad \mathbf{x}^{(s)} = \{\mathbf{x}^{(s)}_{[b]}, \tilde{\mathbf{x}}^{(s)}\} \tag{5.13}$$

The negative log-likelihood can then be written[3]:

$$\mathcal{L}(\theta) = -\log\left[\sum_{\kappa=1}^{N_k} \hat{w}_\kappa(\theta) \prod_{(\beta,\tau)\in[b][t+1:t+h]} \left(\frac{(\hat{\lambda}_{\beta,\kappa,\tau}(\theta))^{y_{\beta,\tau}} \exp\{-\hat{\lambda}_{\beta,\kappa,\tau}(\theta)\}}{(y_{\beta,\tau})!}\right)\right] \tag{5.14}$$

This is the same expression as the joint probability mass function in Equation (5.7) but parametrized as a function of the neural network parameters θ. The maximum likelihood estimation method (MLE) has desirable properties like statistical efficiency and consistency. However, the mixture components cannot be estimated separately, for this reason, MLE is only feasible for hierarchical time series with a small number of series and needs its scalability improved.

Maximum Composite Likelihood

An attractive, computationally efficient alternative to MLE for estimating the model parameters is to maximize the composite likelihood. This method involves breaking up the high-dimensional space into smaller sub-spaces, and the composite likelihood consists of the weighted product of the marginal likelihoods of the subspaces (Lindsay, 1988). For simplicity, we used uniform weights. In addition to the computational efficiency, maximizing the composite likelihood provides a robust and unbiased estimate of marginal model parameters with the drawback that the model inference may suffer from properties similar to a misspecified model (Varin et al., 2011). We will discuss variants of the composite likelihood below.

[3]We kept notations simple and omitted the explicit conditioning on input features.

Naive Bottom Up: A simple option of using composite likelihood is to define each bottom-level time series as its likelihood sub-space and treat them as independent during model training (Orcutt et al., 1968). We refer to this estimation method for the `HINT` as *Naive Bottom Up* (`HINT-NaiveBU`). The negative logarithm of the `HINT-NaiveBU` composite likelihood follows:

$$\mathcal{L}_{\text{NaiveBU}}(\theta) = -\sum_{\beta\in[b]} \log \left[\sum_{\kappa=1}^{N_k} \hat{w}_{\kappa}(\theta) \prod_{\tau\in[t+1:t+h]} \left(\frac{\left(\hat{\lambda}_{\beta,\kappa,\tau}(\theta)\right)^{y_{\beta,\tau}} \exp\left\{-\hat{\lambda}_{\beta,\kappa,\tau}(\theta)\right\}}{(y_{\beta,\tau})!} \right) \right] \tag{5.15}$$

Even though the sub-space consists of single bottom level time series, `HINT-NaiveBU` is still a multi-variate model with the composite likelihood defined over multiple time points $[t+1 : t+h]$. Maximizing the `HINT-NaiveBU` composite likelihood will still learn correlations across the time points and will generate coherent forecast distributions for aggregations in the time dimension. It does not attempt, however, to discover correlations across different time series.

Group Bottom Up: If prior information helps us identify groups of time series with interesting correlation structures, we may estimate them by including the groups in the composite likelihood. We refer to this estimation method for the `HINT` as *Group Bottom Up* (`HINT-GroupBU`). Let $\mathcal{G} = \{[g_i]\}$ be time-series groups, then the negative log composite likelihood for the `HINT-GroupBU` is

$$\mathcal{L}_{\text{GroupBU}}(\theta) = -\sum_{[g_i]\in\mathcal{G}} \log \left[\sum_{\kappa=1}^{N_k} \hat{w}_{\kappa}(\theta) \prod_{(\beta,\tau)\in[g_i][t+1:t+h]} \left(\frac{\left(\hat{\lambda}_{\beta,\kappa,\tau}(\theta)\right)^{y_{\beta,\tau}} \exp\left\{-\hat{\lambda}_{\beta,\kappa,\tau}(\theta)\right\}}{(y_{\beta,\tau})!} \right) \right] \tag{5.16}$$

The main advantage over `HINT-NaiveBU` is that the model now learns to capture the cross-series relationships. In this paper we only rely on intuitive grouping like geographic proximity, but one could in principle employ more sophisticated methods like clustering to define the groups, as we mention in Section 5.6. To optimize the learning objective we use stochastic gradient descent, and sample train series batches at the group level.

Forecast Inference

As mentioned earlier, model inference from composite likelihood estimation suffers from problems similar to model misspecification. This is because of the independence assumed across sub-spaces. In our model, the maximum composite likelihood estimates do not understand how mixture components learnt for one sub-space relate to mixture components learnt for a different subspace. However, we need to identify mixture components across sub-spaces in order to define the joint distribution in Equation (5.7) across all bottom level time series. Knowing this joint distribution is at the crux of forecast inference from our model. Fortunately, there is a natural way for us to resolve this issue. Both in the `HINT-NaiveBU` composite likelihood in Equation (5.15) and in the `HINT-GroupBU` composite likelihood in Equation (5.16), the weights $\hat{\mathbf{w}}_{[k]}(\theta) \in \mathbb{R}^{N_k}\}$ are shared across all sub-spaces. . We identify components with the same weight as belonging to the same multivariate sample, and hence providing the full joint distribution. We call this method *weight matching*.

For the `HINT-NaiveBU` estimation method, the weight matching method is an statistical model over extension on the simplicity of the joint distribution. The `HINT-GroupBU` approach significantly alleviates this problem because the model parameters are well defined within each group $[g_i] \in \mathcal{G}$ and if most of the interesting correlations are already captured within each group, then much less burden is placed on the weight matching method. We show in the empirical evaluation of Section Section 5.5 that both `HINT-NaiveBU` and `HINT-GroupBU` models perform favorably when compared to other mean and probabilistic hierarchical forecasting methods, and between the two, `HINT-GroupBU` is generally more accurate when the time series grouping given is informative.

5.4.6 Hierarchical Mixture Network

Our primary goal is to create a probabilistic coherent forecasting model that is accurate and efficient. For this purpose, we opt to extend the MQForecaster family (Wen et al., 2017; Eisenach et al., 2021), proven by its history of industry service, with the Poisson mixture distribution. We refer to this model as the *Hierarchical Forecast Network* (`HINT`). Our `MQForecaster` architecture selection is driven by its high computational efficiency consequence of the *forking-sequences* technique and multi-step forecasting strategy. In addition to its ability to incorporate static, and known future temporal features.

5.4.7 Model Features

As part of the innovations within our work, we propose to separate the bottom-level and aggregate-level features that we use in the forecasts. Sharing aggregate-level features across their respective bottom series, helps simplify the model's inputs and reducing redundant information, and greatly improving the model's memory requirements.

We follow the hierarchical forecasting literature practice for the *static features* and send the group identifiers implied by the hierarchy structure. We denote them

$$\mathbf{x}^{(s)} = \{\mathbf{x}^{(s)}_{[b]},\ \tilde{\mathbf{x}}^{(s)}\}. \tag{5.17}$$

Regarding *temporal features*, for the bottom level, we use the bottom series' past; for the aggregate level we use the parent node series' past. The future temporal information available can be problem specific like prices or promotions, or other simpler model forecasts as inputs, such as Naive or SNaive that help the model predict series levels and seasonalities. We denote the historical and future temporal features as

$$\mathbf{x}^{(h)}_{[:t]} = \{\mathbf{x}^{(h)}_{[b][:t]},\ \tilde{\mathbf{x}}^{(h)}_{[:t]}\} \qquad \text{and} \qquad \mathbf{x}^{(f)}_{[:t+h]} = \{\mathbf{x}^{(f)}_{[b][:t+h]},\ \tilde{\mathbf{x}}^{(f)}_{[:t+h]}\} \tag{5.18}$$

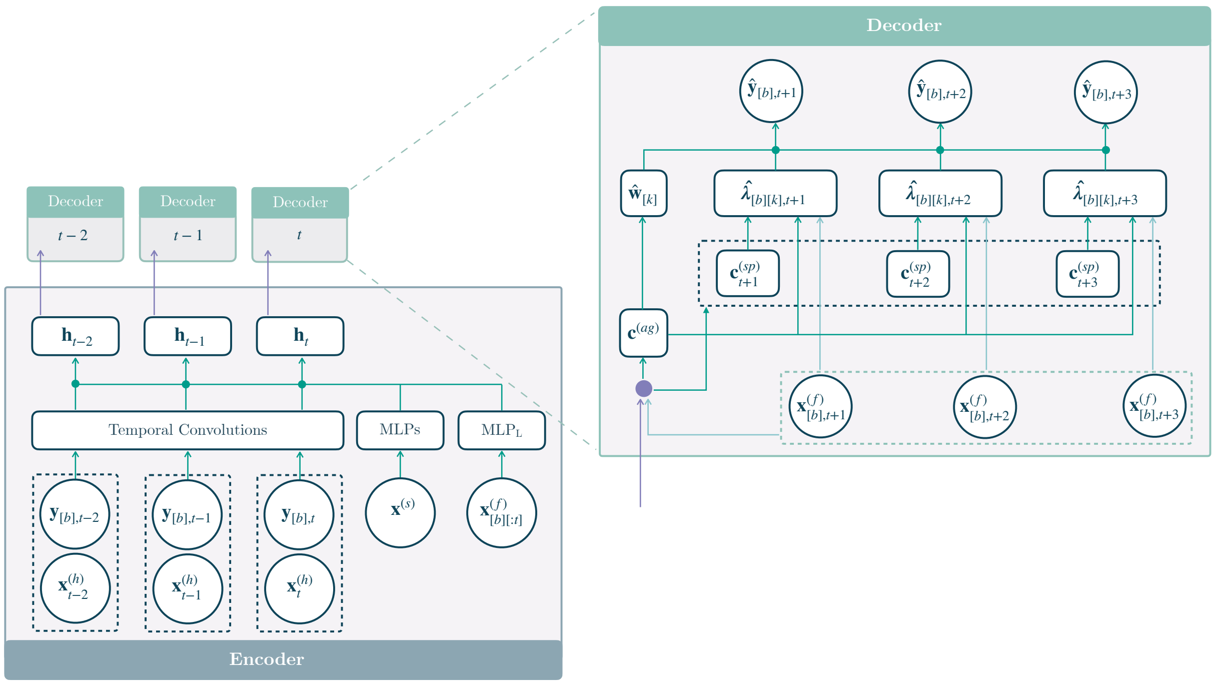

Decoder
$\hat{\mathbf{y}}_{[b],t+1}$
$\hat{\mathbf{y}}_{[b],t+2}$
$\hat{\mathbf{y}}_{[b],t+3}$
$\hat{\mathbf{w}}_{[k]}$
$\hat{\boldsymbol{\lambda}}_{[b][k],t+1}$
$\hat{\boldsymbol{\lambda}}_{[b][k],t+2}$
$\hat{\boldsymbol{\lambda}}_{[b][k],t+3}$
$\mathbf{c}^{(sp)}_{t+1}$
$\mathbf{c}^{(sp)}_{t+2}$
$\mathbf{c}^{(sp)}_{t+3}$
$\mathbf{c}^{(ag)}$
$\mathbf{x}^{(f)}_{[b],t+1}$
$\mathbf{x}^{(f)}_{[b],t+2}$
$\mathbf{x}^{(f)}_{[b],t+3}$
Decoder $t-2$
Decoder $t-1$
Decoder t
$\mathbf{h}_{t-2}$
$\mathbf{h}_{t-1}$
$\mathbf{h}_{t}$
Temporal Convolutions
MLPs
MLP$_\text{L}$
$\mathbf{y}_{[b],t-2}$
$\mathbf{y}_{[b],t-1}$
$\mathbf{y}_{[b],t}$
$\mathbf{x}^{(s)}$
$\mathbf{x}^{(f)}_{[b][:t]}$
$\mathbf{x}^{(h)}_{t-2}$
$\mathbf{x}^{(h)}_{t-1}$
$\mathbf{x}^{(h)}_{t}$
Encoder

Figure 5.3: The *Hierarchical Forecast Network* (HINT) is a *Sequence-to-Sequence with Context* network that uses dilated temporal convolutions as the primary encoder and MLP decoders for a direct multi-step forecast. The forked decoders share parameters and create joint forecast distribution for each encoder's time point, making the architecture efficient in its optimization and forecasts.

5.4.8 Model Architecture

In summary, the HINT builds on the Multi Quantile Forecaster Family (MQForecaster) architecture that is based on Seq2Seq-C (Cho et al., 2014). The HINT uses Temporal Convolution Network (TCN) (Oord et al., 2016) to encode the available history into hidden states and uses forked decoders based on Multi Layer Perceptron (MLP) (Rosenblatt, 1961) in a direct multi-horizon forecast strategy (Amir and Souhaib, 2016).

Encoder

As explained earlier the HINT main encoder is a stack of dilated temporal convolutions. Additionally, we use a global dense layer to encode the static features and a local dense layer, shared across time, to encode the available future information. The encoder for each time $[t]$ and its components are described in Equation (5.19).

$$\begin{aligned}
\mathbf{h}^{(h)}_t &= \{\mathbf{h}^{(h)}_{1,t}\,,\, \tilde{\mathbf{h}}^{(h)}_{2,t}\} = \{\mathbf{TCN}(\mathbf{x}^{(h)}_{[b][:t]})\,,\, \mathbf{TCN}(\tilde{\mathbf{x}}^{(h)}_{[:t]})\} \\
\mathbf{h}^{(s)} &= \{\mathbf{h}^{(s)}_1\,,\, \mathbf{h}^{(s)}_2\} = \{\mathbf{MLP}(\mathbf{x}^{(s)}_{[b]})\,,\, \mathbf{MLP}(\tilde{\mathbf{x}}^{(s)})\} \\
\mathbf{h}^{(f)}_t &= \mathbf{MLP}_L(\mathbf{x}^{(f)}_{[b][:t+h]})
\end{aligned} \tag{5.19}$$

The encoder's output in Equation (5.20) is a set of shared and bottom level encoded features $\mathbf{h}_{1,t}$ and $\mathbf{h}_{2,t}$. The first concatenates all the encoded past $\mathbf{h}^{(h)}_{1,t}$, $\mathbf{h}^{(h)}_{2,t} \in \mathbb{R}^{N_{cf}}$, static $\mathbf{h}^{(s)}_{1}$, $\mathbf{h}^{(s)}_{2} \in \mathbb{R}^{N_s}$ and available future $\mathbf{h}^{(f)}_{t} \in \mathbb{R}^{N_f}$ information[4]. The second one concatenates the encoded past $\mathbf{h}^{(p)}_{2,t} \in \mathbb{R}^{N_p}$, and static $\mathbf{h}^{(s)}_{2} \in \mathbb{R}^{N_s}$ shared features.

$$\mathbf{h}_t = \{\mathbf{h}_{1,t},\ \mathbf{h}_{2,t}\} = \{[\mathbf{h}^{(h)}_{1,t}|\mathbf{h}^{(h)}_{2,t}|\mathbf{h}^{(s)}_{1}|\mathbf{h}^{(s)}_{2}|\mathbf{h}^{(f)}_{t}],\quad [\mathbf{h}^{(h)}_{2,t}|\mathbf{h}^{(s)}_{2}]\} \tag{5.20}$$

Forked Decoders

The HINT uses a two-branch MLP decoder. The first decoder branch, summarizes the encoder output and future available information into two contexts: The horizon-agnostic context set $\mathbf{c}^{(ag)} \in \mathbb{R}^{N}_{ag}$ and the horizon-specific context $\mathbf{c}^{(sp)}_{[t+1:t+h]} \in \mathbb{R}^{N_{sp}\times h}$ that provides structural awareness of the forecast horizon and plays a crucial role in expressing recurring patterns in the time series if any. Equation (5.21) describes the first decoder branch:

$$\begin{aligned} \mathbf{c}^{(ag)} &= \{\mathbf{c}^{(ag)}_{1},\ \mathbf{c}^{(ag)}_{2}\} = \{\mathbf{MLP}(h_{1,t}),\ \mathbf{MLP}(h_{2,t})\} \\ \mathbf{c}^{(sp)}_{[t+1:t+h]} &= \mathbf{MLP}_{L}(h_{1,t}) \end{aligned} \tag{5.21}$$

The second decoder branch adapts the horizon-specific and horizon-agnostic contexts into the parameters of the Poisson mixture distribution. For the horizon-specific Poisson rates, we use the forking-sequence technique with a series of decoders with shared parameters for each time point $[t]$ and for the mixture weights, we apply an MLP followed by a softmax on the aggregate horizon agnostic context. Equation (5.22) describes the second decoder branch:

$$\begin{aligned} \hat{\boldsymbol{\lambda}}_{[b][k][t+1:t+h]} &= \mathbf{MLP}_{L}(\mathbf{c}^{(ag)}_{1},\ \mathbf{c}^{(sp)}_{[t+1:t+h]},\ \mathbf{x}^{(f)}_{[b][t+1:t+h]}) \\ \hat{\mathbf{w}}_{[k]} &= \mathrm{SoftMax}(\mathbf{MLP}(\mathbf{c}^{(ag)}_{2})) \end{aligned} \tag{5.22}$$

5.5 Experiments

5.5.1 Hierarchical Datasets

To evaluate our method, we consider three forecasting tasks where the objective is to provide quantile forecasts for each time series in the group or hierarchy structure. All the three datasets that we use in the empirical evaluation [5] are publicly available and have been used in the hierarchical forecasting literature (Wickramasuriya et al., 2019; Souhaib and Bonsoo, 2019; Rangapuram et al., 2021; Paria et al., 2021). Table 5.1 summarizes the datasets' characteristics[6]

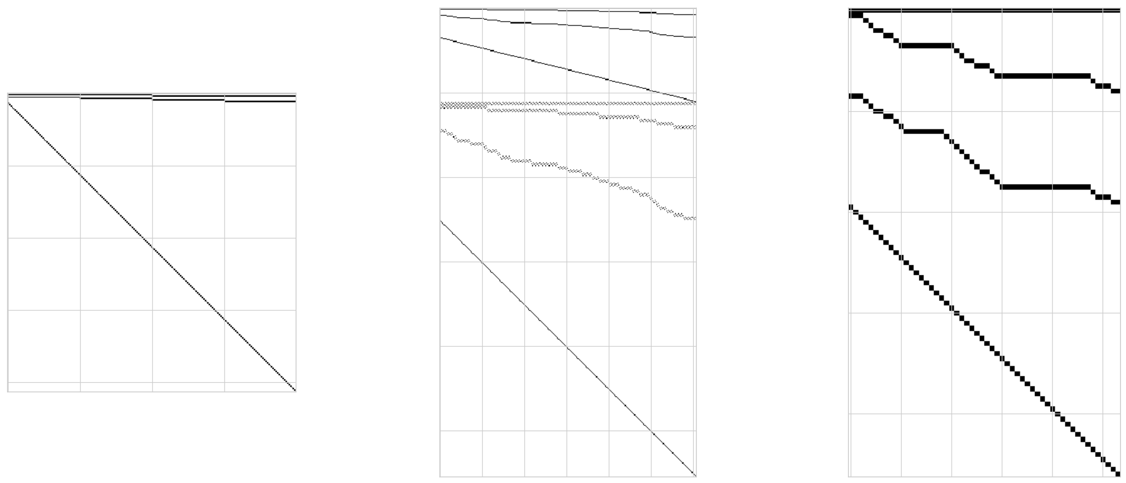
[4]The local horizon-specific MLP$_L$ aligns future seasonalities and events and improves the forecast's sharpness.

[5]`Traffic` is available at the UCI ML repository. `Tourism-L` is available at `MinT` reconciliation web page. `Favorita` is available in its Kaggle Competition url.

[6]We include more details for the `Traffic`, `Tourism-L`, and `Favorita` datasets in Appendix B.4.

Table 5.1: Summary, hierarchical structure and forecast horizon of datasets used in our empirical study.

Dataset	Total	Aggregated	Bottom	Levels	Observations	Horizon (h)
`Traffic`	207	7	200	4	366	1
`Tourism-L`	555	175	76 / 304	4 / 5	228	12
`Favorita`	371,312	153,386	217,944	4	1,688	34

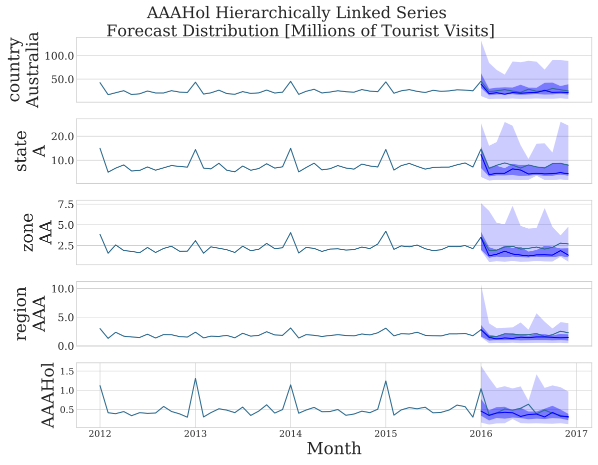


(a) Traffic **(b)** Tourism **(c)** Favorita

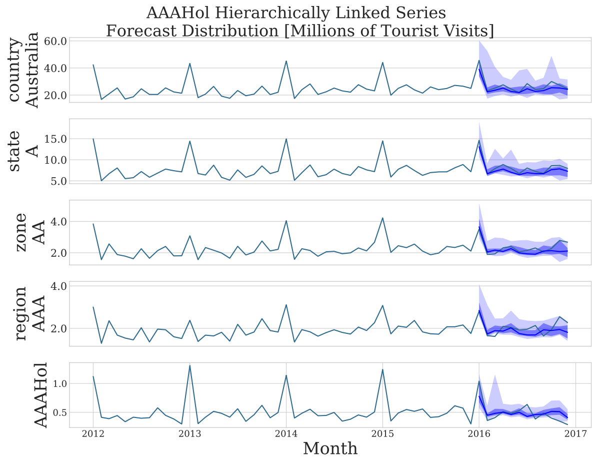
Figure 5.4: Empirical evaluation datasets' hierarchical constraints visualization. (a) `Traffic` groups 200 highways' occupancy series into quarters, halves and total. (b) `Tourism-L` groups its 555 regional visit series, into a combination travel purpose, zones, states and country geographical aggregations. (c) `Favorita` groups its grocery sales geographically, by store, city, state and country levels.

The `Tourism-L` (Tourism Australia, Canberra, 2019) is an Australian Tourism dataset that contains 555 monthly visit series from 1998 to 2016, grouped by geographical regions and travel purposes. `Favorita` (Corporación Favorita, 2018) is a Kaggle competition dataset of grocery item sales daily history with additional information on promotions, items, stores, and holidays, containing 371,312 series from January 2013 to August 2017, with a geographic hierarchy of states, cities, and stores. We show their hierarchical constraints matrix in Figure 5.4. `Traffic` (Dua and Graff, 2017; Souhaib and Bonsoo, 2019) measures the occupancy of 200 car lanes in the San Francisco Bay Area freeways, randomly grouped into a year of daily observations with 207 series hierarchical structure.

The datasets unique characteristics provide an opportunity to showcase the broad applicability of the `HINT`. `Tourism-L` allows us to test the `HINT` to model group structures with multiple hierarchies. `Favorita` allows us to test the `HINT` on a large-scale dataset. `Traffic` is composed of randomly assigned hierarchical groupings that may not have any informative structures for the `HINT` to learn with `GroupBU`.



(a) HINT-NaiveBU



(b) HINT-GroupBU

Figure 5.5: HINT-NaiveBU and HINT-GroupBU forecast distributions on a Tourism-L hierarchical series. The top row shows total tourist visits in Australia, the second row shows the visits to Australia for the North South Wales state (A), the third row shows the holiday visits in the metropolitan Area of New South Wales (AA), the fourth row shows the Sidney total visits (AAA), the final row shows the holiday visits to Sidney. Forecast distributions, 99% and 75% prediction intervals in light and dark blue.

5.5.2 Time Series Covariance Modeling

As described in Section 5.4.4, when in the presence of informative group structures, the HINT can use the multivariate Poisson joint distribution expressiveness to its advantage, as it better models the dependencies within the series groups considered in its estimation while remaining computationally efficient. In this subsection, we demonstrate HINT's distribution versatility showing how it can leverage the presence of complex correlation structures like the ones present Tourism-L and Favorita datasets to improve the forecast sharpness.

Figure 5.5 shows how HINT-NaiveBU's composite likelihood suffers from a joint bottom-level series probability misspecification, as it does not consider its interactions. In contrast, the HINT-GroupBU method correctly estimates Tourism-L's bottom-level correlations improving the forecast distribution concentration of the upper-level series. We validate these intuitions in the experiments of Section 5.5 where HINT-GroupBU outperforms the probabilistic coherent forecasting baselines in the Tourism-L and Favorita datasets with informative group series structures. HINT-NaiveBU performs well on disaggregated series and means as we show in Section 5.5, and produces forecasts comparable to those of hierarchical forecasting baselines on the Tourism-L and Favorita datasets, while it outperforms them in the Traffic dataset where the hierarchical structure is noisy or uninformative, since its group structure was randomly assigned.

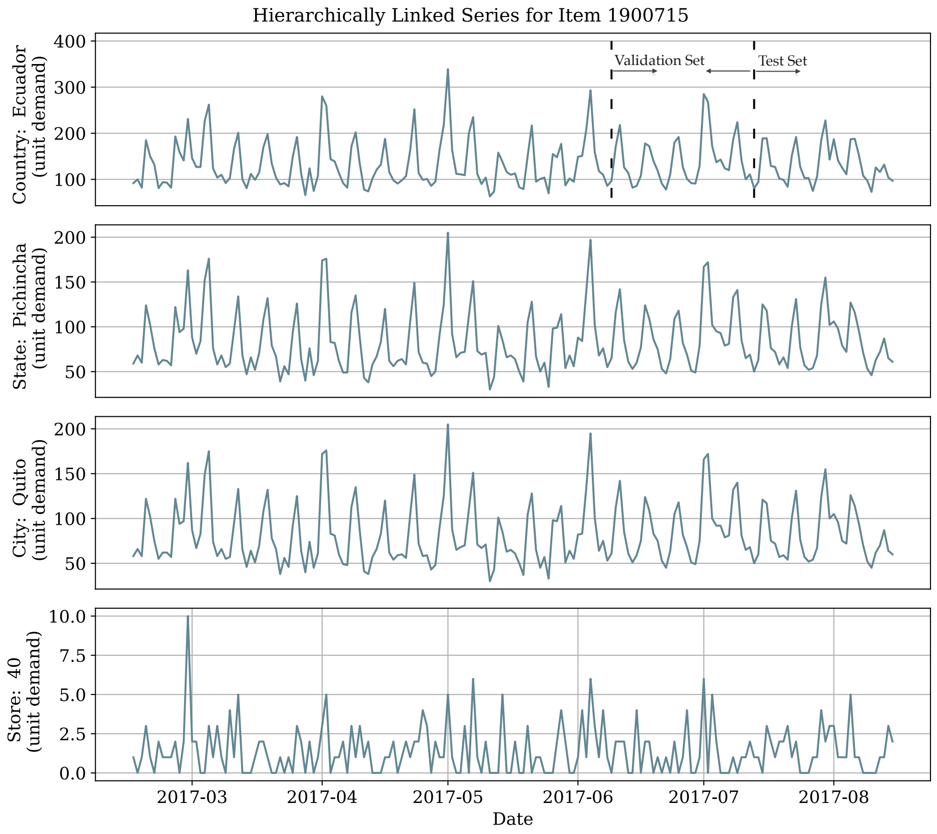


Figure 5.6: Geographically linked time series from the `Favorita` dataset. The top level shows the sales for a grocery item for the Favorita stores in the country of Ecuador. The second level shows the sold units within the Pichincha state, the third level shows the sales for the city of Quito, the final level shows the sales for the item at a particular store. For this dataset, the training set comprises all the observations preceding the validation and test sets. The validation set (between the first and second dotted lines) is the 34 days before the test set. The held-out test set (marked by the last dotted line) is the last 34 observations.

5.5.3 Datasets Partition and Preprocessing

For the main experiments we separate the train, validation and test datasets' *partition* as follows: we hold out the final horizon-length observations as the *test set*. In a sliding-window fashion, we use the horizon-length that precedes the *test set* as the *validation set* and treat as *training set* the rest of the past information. A *partition* example is depicted in Figure 5.6.

For comparability with recent hierarchical forecasting literature, we keep ourselves as close as possible to the preprocessing and wrangling of the datasets to that of Rangapuram et al., 2021[7]. In general, the *static* variables that we consider on all the datasets correspond to the hierarchical and group designators as categorical variables implied by the hierarchical constraint matrix. The *temporal* covariates that we consider are the time series for the upper levels of the hierarchy, as well as calendar covariates associated with the time series frequency of each dataset. As *future* data, we include calendar covariates to help the `HINT` capture seasonalities.

[7]The pre-processed datasets are available in the hierarchical forecasting extension to the `GluonTS` library.

5.5.4 Evaluation

Probabilistic Forecasting

The primary evaluation metric of the model's forecasts is based on the Multi Quantile Loss (MQL) (Matheson and Winkler, 1976); consider the estimated cumulative distribution function $\hat{F}_{i,\tau}$, of the variable $Y_{i,\tau}$, and its observation $y_{i,\tau}$, the loss is defined as:

$$\mathrm{QL}(\hat{F}_{i,\tau}, y_{i,\tau})_q = 2\left(\mathbb{1}\{y_{i,\tau} \leq \hat{F}^{-1}_{i,\tau}(\,q\,)\} - q\right)\left(\hat{F}^{-1}_{i,\tau}(\,q\,) - y_{i,\tau}\right) \tag{5.23}$$

We summarize the evaluation, for convenience of exposition and to ensure the comparability of our results with the existing literature, using the Continuous Ranked Probability Score (CRPS) (Matheson and Winkler, 1976)[8]. We use the following mean scaled CRPS (Bolin and Wallin, 2019; Makridakis et al., 2022) version:

$$\mathrm{CRPS}(\hat{F}_{[i],\tau}, \mathbf{y}_{[i],\tau}) = \frac{2}{|[\,i\,]|}\sum_{i}\int_{0}^{1} \mathrm{QL}(\hat{F}_{i,\tau}, y_{i,\tau})_q dq \tag{5.24}$$

$$\mathrm{sCRPS}(\hat{F}_{[i],\tau}, \mathbf{y}_{[i],\tau}) = \frac{\mathrm{CRPS}(\hat{F}_{[i],\tau}, \mathbf{y}_{[i],\tau})}{\sum_i |y_{i,\tau}|} \tag{5.25}$$

The CRPS measures the forecast distributions' accuracy and has desirable theoretical properties as a metric (Gneiting and Ranjan, 2011; Bolin and Wallin, 2019). For instance it is a *proper scoring rule*, since for any forecast distribution $\hat{F}_{i,\tau}$ and true distribution $F_{i,\tau}$ the expected score satisfies:

$$\mathbb{E}_{Y_{i,\tau}}\left[\mathrm{CRPS}(F_{i,\tau}, Y_{i,\tau})\right] \leq \mathbb{E}_{Y_{i,\tau}}\left[\mathrm{CRPS}(\hat{F}_{i,\tau}, Y_{i,\tau})\right] \tag{5.26}$$

which implies that it will prefer an ideal probabilistic forecasting system over any other.

The main focus of the paper is probabilistic coherent forecasting, yet we complement the results from Section 5.5.6 with the evaluation of HINT's mean hierarchical forecasts in Section 5.5.7.

Mean Forecasting

To evaluate the mean hierarchical forecasts we take recommendations from Hyndman and Koehler, 2006 and define the relative Mean Squared Error (relMSE) based on the following Equation:

$$\mathrm{relMSE}(\mathbf{y}, \hat{\mathbf{y}}, \tilde{\mathbf{y}}) = \frac{\mathrm{MSE}(\mathbf{y}, \hat{\mathbf{y}})}{\mathrm{MSE}(\mathbf{y}, \tilde{\mathbf{y}})} \tag{5.27}$$

where $\mathbf{y}, \hat{\mathbf{y}}, \tilde{\mathbf{y}} \in \mathbb{R}^{N\times H}$ represent the time series observations, the mean forecasts and the Naive baseline forecasts respectively. As a relative measurement relMSE removes the data's scale while comparing to the baseline prediction.

[8]In practice the evaluation of the CRPS uses numerical integration technique, that discretizes the quantiles and treats the integral with a left Riemann approximation, averaging over uniformly distanced quantiles.

Table 5.2: Considered hyperparameters for the *Hierarchical Forecast Network* (HINT). The learning rate, random seed, and SGD epochs that performed best on the validation set were selected automatically in each HYPEROPT run. The rest parameters were configured once per dataset, as explained in B.6.

* The Parametrized Exponential Linear Unit (PeLU) modifies the ReLU activation improving the network's training speed Clevert et al., 2015.

Hyperparameter	Considered Values
Initial learning rate for SGD optimization.	$\text{lr} \in [0.00001, 0.01]$
SGD full passes to dataset (epochs).	$\text{n\_epochs} \in \{10, \ldots, 3000\}$
Random seed that controls initialization of weights.	$\text{seed\_train} \in \{1, \ldots, 10\}$
SGD Batch Size.	$\text{batch\_size} \in [4, 100]$
Activation Function.	PeLU*
Temporal Convolution Kernel Size.	$N_{ck} \in \{2,\ 7\}$
Temporal Convolution Layers.	$N_{cl} \in \{3,\ 5\}$
Temporal Convolution Filters.	$N_{cf} \in \{10,\ 30\}$
Future Encoder Dimension.	$N_{f} \in \{50\}$
Static Encoder Dimension.	$N_{s} \in \{100\}$
Horizon Agnostic Decoder Dimensions.	$N_{ag} \in \{50\}$
Horizon Specific Decoder Dimensions.	$N_{sp} \in \{20\}$
Poisson Mixture Weights Decoder Layers.	$N_{wdl} \in \{3, 4\}$
Poisson Mixture Rate Decoder Layers.	$N_{rdl} \in \{2, 3, 4\}$
Local Decoder Dimensions.	$N_{k} \in \{25,\ 50,\ 100\}$

5.5.5 Train and Hyperparameter Optimization

For the overall hyperparameter selection, we used a standard two-stage approach where we first fixed the architecture, and the estimated probability distribution, and a second stage where we optimized the architecture's training procedure. Keeping a second stage explored hyperparameter space small serves two purposes: It keeps space exploration computationally tractable and showcases HINT's robustness, broad applicability, and accuracy with minor modifications. We defer some hyperparameter selection details to Appendix B.6.

In the first stage we select the number of HINT's mixture components that are responsible for single-series forecasting and modeling bottom-level correlations, as stated in Section 5.4.5 and shown in Appendix B.3. For each dataset, we selected the components optimally using temporal cross-validation in Appendix B.5 ablation study, where we found that complex correlation structures favored a higher number of components. To observe the effects of modeling the series covariance, we compared HINT-GroupBU and HINT-NaiveBU variants.

In the second stage, as shown in Table 5.2, the hyperparameter space that we consider for optimization is minimal. We only tune the learning rate, random seed to escape underperforming local minima, and the number of SGD epochs as a form of regularization (Yao et al., 2007). During the *hyperparameter optimization phase*, we measure the model sCRPS performance on the validation set described in Section 5.5.3, and use HYPEROPT (Bergstra et al., 2011), a Bayesian optimization library, to efficiently explore the hyperparameters based on the validation measurements.

After the optimal hyperparameters are determined, we estimate the model parameters again by shifting the training window forward, noted as the *retrain phase*, and predict for the final test set. We refer to the combination of the *hyperparameter optimization* and *retrain* phases as a *run*. The HINT is implemented using MXNet (Tianqi Chen et al., 2015). To train the network, we minimize the negative log-composite likelihood variant from Section 5.4.5, using stochastic gradient descent with Adaptive Moment Estimation SGD (ADAM) (Kingma and Ba, 2014).

Table 5.3: Empirical evaluation of probabilistic coherent forecasts. Mean *scaled continuous ranked probability score* (sCRPS) averaged over 8 runs, at each aggregation level, the best result is highlighted (lower measurements are preferred). Methods without standard deviation have deterministic solutions.
* The HierE2E results differ from Rangapuram et al., 2018, sCRPS quantile interval space has granularity of 1 percent over its original 5 percent.
** PERMBU-MinT on Tourism-L is unavailable because the original implementation, currently can't be applied to structures beyond single hierarchies.

Dataset	Level	HINT-GroupBU	HINT-NaiveBU	HierE2E*	PERMBU-MinT**	ARIMA	GLMPoisson
Traffic	Overall	0.0907 ± 0.0024	0.0704 ± 0.0014	**0.0375 ± 0.0058**	0.0677 ± 0.0061	0.0751	0.0771
	1 (geo.)	0.0397 ± 0.0044	**0.0134 ± 0.0022**	0.0183 ± 0.0091	0.0331 ± 0.0085	0.0376	0.0063
	2 (geo.)	0.0537 ± 0.0024	0.0289 ± 0.0017	**0.0183 ± 0.0081**	0.0341 ± 0.0081	0.0412	0.0194
	3 (geo.)	0.0538 ± 0.0022	0.0290 ± 0.0011	**0.0209 ± 0.0071**	0.0417 ± 0.0061	0.0549	0.0406
	4 (geo.)	0.2155 ± 0.0022	0.2101 ± 0.0008	**0.0974 ± 0.0021**	0.1621 ± 0.0027	0.1665	0.2420
Tourism-L	Overall	**0.1249 ± 0.0020**	0.1274 ± 0.0028	0.1472 ± 0.0029	-	0.1416	0.1762
	1 (geo.)	**0.0431 ± 0.0042**	0.0514 ± 0.0030	0.0842 ± 0.0051	-	0.0263	0.0854
	2 (geo.)	**0.0637 ± 0.0032**	0.0705 ± 0.0026	0.1012 ± 0.0029	-	0.0904	0.1153
	3 (geo.)	0.1084 ± 0.0033	**0.1068 ± 0.0019**	0.1317 ± 0.0022	-	0.1389	0.1691
	4 (geo.)	0.1554 ± 0.0025	**0.1507 ± 0.0014**	0.1705 ± 0.0023	-	0.1878	0.2165
	5 (prp.)	**0.0700 ± 0.0038**	0.0907 ± 0.0061	0.0995 ± 0.0061	-	0.0770	0.0954
	6 (prp.)	**0.1070 ± 0.0023**	0.1175 ± 0.0047	0.1336 ± 0.0042	-	0.1270	0.1682
	7 (prp.)	0.1887 ± 0.0032	**0.1836 ± 0.0038**	0.1955 ± 0.0025	-	0.2022	0.2458
	8 (prp.)	0.2629 ± 0.0034	**0.2481 ± 0.0026**	0.2615 ± 0.0016	-	0.2834	0.3134
Favorita	Overall	**0.4020 ± 0.0182**	0.5301 ± 0.0120	0.5298 ± 0.0091	0.4670 ± 0.0096	0.4373	0.4524
	1 (geo.)	0.2760 ± 0.0149	0.4166 ± 0.0195	0.4714 ± 0.0103	**0.2692 ± 0.0076**	0.3112	0.3611
	2 (geo.)	0.3865 ± 0.0207	0.5128 ± 0.0108	0.5182 ± 0.0107	**0.3824 ± 0.0092**	0.4183	0.4398
	3 (geo.)	**0.4068 ± 0.0206**	0.5317 ± 0.0115	0.5291 ± 0.0129	0.6838 ± 0.0108	0.4446	0.4598
	4 (geo.)	**0.5387 ± 0.0253**	0.6594 ± 0.0150	0.6012 ± 0.0131	0.5532 ± 0.0116	0.5749	0.5490

5.5.6 Forecasting Results

Probabilistic Forecasting Results

For our proposed methods, we report the HINT-NaiveBU and the HINT-GroupBU. As described in Section 5.4.5, the HINT-NaiveBU treats the bottom level series as independent observations, and the HINT-GroupBU considers groups of time series during its composite likelihood estimation. Both methods obtain probabilistic coherent forecasts using the bottom-up reconciliation. The comparison of the HINT variants serves as an ablation experiment to better analyze the source of the accuracy improvements. It also showcases the ability of the Poisson Mixture model to give good results for unseen hierarchical structures, and in the case of the Traffic dataset, of uninformative or noisy time-series group structure, to explore the limits of the GroupBU estimation method.

Table 5.3 shows sCRPS measurements for predictive distributions at each level of the dataset hierarchy. The overall sCRPS score is reported in the top row, with the best result highlighted in bold. The HINT improves the overall sCRPS for Tourism-L and Favorita. The HINT-GroupBU variant significantly shows 11.8% percent and 8.1% percent, respectively. However, in the Traffic dataset, the HINT-GroupBU variant does not improve upon other methods. We hypothesize that holiday features explain the Traffic New Year's day performance gap between HierE2E's and alternative approaches. The HINT-NaiveBU variant performs well on Traffic and gives an acceptable performance on Tourism-L and Favorita.

Table 5.4: Empirical evaluation of mean hierarchical forecasts. *Relative Mean Squared Error* (relMSE) averaged over 8 runs, at each aggregation level, the best result is highlighted (lower measurements are preferred). Methods without standard deviation have deterministic solutions.
[*] The ARIMA-ERM results for `Tourism-L` differ from Rangapuram et al., 2021, as we improved the numerical stability of their implementation.

Dataset	Level	HINT-GroupBU	HINT-NaiveBU	ARIMA-ERM[*]	ARIMA-MinT-ols	ARIMA	GLMPoisson	SNaive
Traffic	Overall	0.1750 ± 0.0099	$\mathbf{0.0168 \pm 0.0026}$	0.0199	0.0425	0.0433	0.0175	0.0709
	1 (geo.)	0.1619 ± 0.0099	$\mathbf{0.0033 \pm 0.0026}$	0.0133	0.0344	0.0302	0.0001	0.0547
	2 (geo.)	0.1835 ± 0.0101	0.0240 ± 0.0027	0.0135	0.0380	0.0392	**0.0109**	0.0676
	3 (geo.)	0.1819 ± 0.0100	$\mathbf{0.0239 \pm 0.0027}$	0.0373	0.0647	0.0850	0.0462	0.0989
	4 (geo.)	0.9964 ± 0.043	0.9561 ± 0.0022	0.6355	0.5876	**0.5669**	1.2119	1.3118
Tourism-L	Overall	$\mathbf{0.1113 \pm 0.0158}$	0.2680 ± 0.0748	0.1178	0.1251	0.1414	0.1944	0.1306
	1 (geo.)	0.0597 ± 0.0212	0.3371 ± 0.1506	0.0596	0.0472	**0.0343**	0.2015	0.0582
	2 (geo.)	$\mathbf{0.1121 \pm 0.0152}$	0.3186 ± 0.1130	0.1293	0.1476	0.2530	0.2274	0.1628
	3 (geo.)	$\mathbf{0.2250 \pm 0.0196}$	0.3909 ± 0.0822	0.2529	0.3556	0.4429	0.3913	0.3695
	4 (geo.)	$\mathbf{0.2980 \pm 0.0197}$	0.4198 ± 0.0668	0.3236	0.4288	0.4835	0.4238	0.4766
	5 (prp.)	0.0798 ± 0.0195	0.1459 ± 0.0177	0.0895	0.0856	0.0973	0.0961	**0.0615**
	6 (prp.)	$\mathbf{0.1403 \pm 0.0150}$	0.1576 ± 0.0113	0.1466	0.1537	0.1663	0.1840	0.1577
	7 (prp.)	$\mathbf{0.2654 \pm 0.0212}$	0.2537 ± 0.0100	0.2705	0.3017	0.2914	0.3293	0.3699
	8 (prp.)	0.3302 ± 0.0235	$\mathbf{0.3030 \pm 0.0083}$	0.3543	0.3970	0.3769	0.3908	0.4969
Favorita	Overall	$\mathbf{0.7563 \pm 0.0713}$	0.9533 ± 0.0201	0.8163	0.9465	0.9665	0.8346	1.1420
	1 (geo.)	$\mathbf{0.7944 \pm 0.0568}$	0.9188 ± 0.0187	0.8362	0.8999	0.9217	0.9054	1.1269
	2 (geo.)	$\mathbf{0.7355 \pm 0.1057}$	1.0451 ± 0.0310	0.7830	1.0057	1.0451	0.8037	1.1078
	3 (geo.)	$\mathbf{0.7303 \pm 0.1035}$	1.0317 ± 0.0333	0.7986	1.0418	1.0881	0.8003	1.1315
	4 (geo.)	$\mathbf{0.6770 \pm 0.0351}$	0.8090 ± 0.0180	0.8199	0.8808	0.8228	0.6499	1.2815

Our results confirm observations from the community that a global model, capable of cross-learning from all the time series jointly, improves the forecasts over those from univariate time series methods. Additionally, the qualitative comparison between the `NaiveBU` and `GroupBU` methods shows that an expressive joint distribution framework capable of leveraging the hierarchical structure of the data, when informative, benefits the forecasts' accuracy.

5.5.7 Mean Hierarchical Forecasting Results

In Section 5.4, the `HINT` joint likelihood defines a probabilistic coherent system for its predictive distributions; the mean hierarchical coherence is naturally implied. In this experiment, we compare `HINT` mean hierarchical forecasts (weighted average of Poisson rates) with the following methods' forecasts: (1) ARIMA-ERM (Souhaib and Bonsoo, 2019) that performs an optimization-based reconciliation free of the unbiasedness assumption of the base forecasts, (2) ARIMA-`MinT` (Wickramasuriya et al., 2019) meant to reconcile unbiased independent forecasts and minimize the variance of the forecast errors, (3) ARIMA-`NaiveBU` (Orcutt et al., 1968) that produces univariate bottom-level time-series forecasts independently and then sums them according to the hierarchical constraints, (4) automatic ARIMA (Hyndman and Khandakar, 2008), (5) `GLMPoisson` (Nelder and Wedderburn, 1972) (6) and the SNaive model.

Table 5.4 contains the relMSE measurements for the predicted means at each aggregation level. The top row reports the overall relMSE (averaged across all the hierarchy levels). We highlight the best result in **bolds**. `HINT` shows overall improvements or comparable results with the baselines'. With respect to mean hierarchical baselines `HINT` shows 4% `Traffic` improvements, 5% `Tourism-L` improvements , and `Favorita` improvements of 7%.

5.6 Conclusion

In this work, we have introduced a novel method for probabilistic coherent forecasting, the *Hierarchical Forecast Network* (HINT), which focuses on learning the joint distribution of bottom level time series and naturally guarantees hierarchical probabilistic coherence. We have also shown through empirical evaluations that our model is accurate for count data. We observed overall significant improvements in sCRPS when compared with previous state-of-the-art probabilistically coherent models on three different hierarchical datasets, Australian domestic tourism (11.8%) and Ecuadorian grocery sales (8.1%). However, the model does not show improvement in sCRPS over alternative approaches when evaluated on San Francisco Bay Area traffic data.

The framework presented here is also extensible. We chose to focus on forecasting count data and used Poisson kernels, but one could also use Gaussian kernels to model joint distributions of real valued hierarchical data. In fact, any kernel which admits closed form expression for aggregated distributions under conditional independence akin to Equation (5.8) will work well, and it includes kernels like the Gamma and the Negative Binomial distributions in addition to the Poisson and the Gaussian distributions already mentioned.

With respect to the definition of the groups considered in the composite likelihood, we selected them naturally inspired by geographic proximity in this work. Still, a promising line of research is an informed creation of such groups based on the series characteristics, for example via clustering.

By formulating the model as a Mixture Density Network, we have separated the probabilistic model of the predictive distribution from the underlying network, making it compatible with any other archiecture. In the current paper we relied on the convolutional encoder version of the MQForecaster architecture, but significant progress has been made in the last few years on neural network based forecasting models; for example, Transformer-based deep learning architectures (Eisenach et al., 2021) that can improve performance. We plan to explore both directions, new kernels and new neural network architectures in future work.

HINT has its drawbacks as well. As is the case with any finite mixture model, the fidelity of the estimated distribution depends on the number of mixture components. A few hundred samples may be sufficient to describe a single marginal distribution but can be too sparse to describe the joint distribution in a high dimensional space. The sparsity will be particularly obvious if customers of hierarchical forecasting are interested in forecast distributions conditioned on partially observed data. The small number of samples will lead to overly confident posterior distributions. Another issue is the model misspecification during inference. The *weight matching* method performs quite well in empirical evaluations but is somewhat unsatisfactory as a statistical model. To mitigate both issues we are exploring generative factor models where the mixture components are truly samples from an underlying distribution and correlations between marginal distributions will be captured by common factors. It will bring HINT closer to standard Hierarchical Bayesian formulation but with fewer and less strict assumptions.

Chapter 6

Long Multi-horizon Forecasting

6.1 Summary

In this case study, we focus on the challenging task of long-horizon forecasting. To tackle the limitations of existing multi-step forecasting strategies, including high volatility and computational complexity, we introduce the *neural hierarchical interpolation for time series* NHITS. Our model addresses these challenges by incorporating innovative hierarchical interpolation and multi-rate data sampling techniques inspired by wavelet analysis. By assembling predictions sequentially and emphasizing components with different frequencies and scales, NHITS significantly improves accuracy in long-horizon forecasting tasks while reducing computation time by orders of magnitude compared to existing neural forecasting approaches.

6.2 Motivation

Long-horizon forecasting is critical in many important applications including risk management and planning. Notable examples include power plant maintenance scheduling (Hyndman and Fan, 2009) and planning for infrastructure construction (Ziel and Steinert, 2018b), as well as early warning systems that help mitigate vulnerabilities due to extreme weather events (Basher, 2006; Field et al., 2012). In healthcare, predictive monitoring of vital signs enables detection of preventable adverse outcomes and application of life-saving interventions (Churpek et al., 2016).

Neural time series forecasting has progressed in a some promising directions. First, the architectural evolution included adoption of the attention mechanism and the rise of Transformer-inspired approaches (Li et al., 2019; Fan et al., 2019; Alaa and Schaar, 2019; Lim et al., 2021), as well as introduction of attention-free architectures composed of deep stacks of fully connected layers (Oreshkin et al., 2020; Olivares et al., 2022a). Both of these approaches are relatively easy to scale up in terms of capacity, compared to LSTMs, and have proven to be capable of capturing long-range dependencies. The attention-based approaches are very generic as they can explicitly model direct interactions between every pair of input-output elements. Unsurprisingly, they happen to be the most computationally expensive. The architectures based on fully connected stacks capture input-output relationships implicitly, however they tend to be more compute-efficient. Second, the recurrent forecast generation strategy has been replaced with the multi-step prediction strategy in both of these approaches. Aside from its convenient bias-variance benefits and robustness (Marcellino et al., 2006; Amir and Souhaib, 2016), the multi-step strategy has enabled the models to efficiently predict long sequences in a single forward pass (Wen et al., 2017; Zhou et al., 2020; Lim et al., 2021).

Despite all the recent progress, long-horizon forecasting remains challenging for neural networks, because their unbounded expressiveness translates directly into *excessive computational complexity* and *forecast volatility*, both of which become especially pronounced in this context. For instance, both attention and fully connected layers scale quadratically in memory and computational cost with respect to the forecasting horizon length. Figure 6.1 illustrates how forecasting errors and computation costs inflate dramatically with growing forecasting horizon in the case of the fully connected architecture electricity consumption predictions. Attention-based predictions show similar behavior.

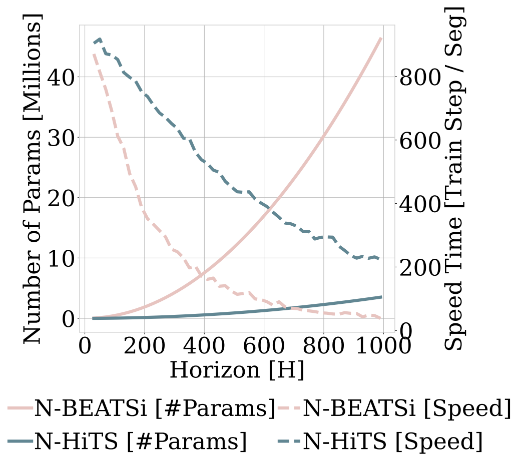
Neural long-horizon forecasting research has mostly focused on attention efficiency making self-attention sparse (Child et al., 2019; Li et al., 2019; Zhou et al., 2020) or local (Li et al., 2019). In the same vain, attention has been cleverly redefined through locality-sensitive hashing (Kitaev et al., 2020) or FFT (Wu et al., 2021). Although that research has led to incremental improvements in compute cost and accuracy, the silver bullet long-horizon forecasting solution is yet to be found. In this paper we make a bold step in this direction by developing a novel forecasting approach that cuts long-horizon compute cost by an order of magnitude while simultaneously offering 16% accuracy improvements on a large array of multi-variate forecasting datasets compared to existing state-of-the-art Transformer-based techniques. We redefine existing fully-connected NBEATS architecture (Oreshkin et al., 2020; Olivares et al., 2022a) by enhancing its input decomposition via multi-rate data sampling and its output synthesizer via multi-scale interpolation. Our extensive experiments show the importance of the proposed novel architectural components and validate significant improvements in accuracy and computational complexity of the proposed algorithm. This chapter's contributions are summarized below:

(i) **Multi-Rate Data Sampling**: We incorporate sub-sampling layers in front of fully-connected blocks, significantly reducing the memory footprint and the amount of computation needed, while maintaining the ability to model long-range dependencies.

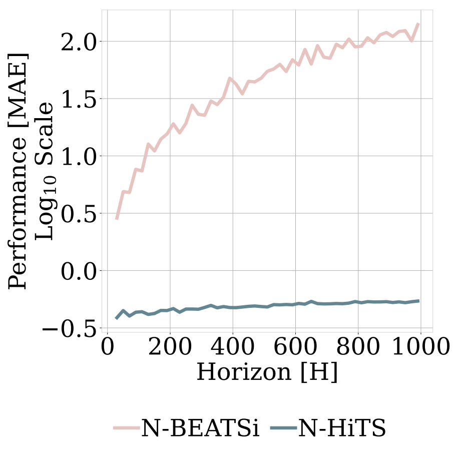
(ii) **Hierarchical Interpolation**: We enforce smoothness of the multi-step predictions by reducing the dimensionality of neural network's prediction and matching its time scale with that of the final output via multi-scale hierarchical interpolation. This novel technique is not unique to our proposed model, and can be incorporated in different architectures.

(iii) **NHITS architecture**: A novel way of hierarchically synchronizing the rate of input sampling with the scale of output interpolation across blocks, which induces each block to specialize on forecasting its own frequency band of the time-series signal.

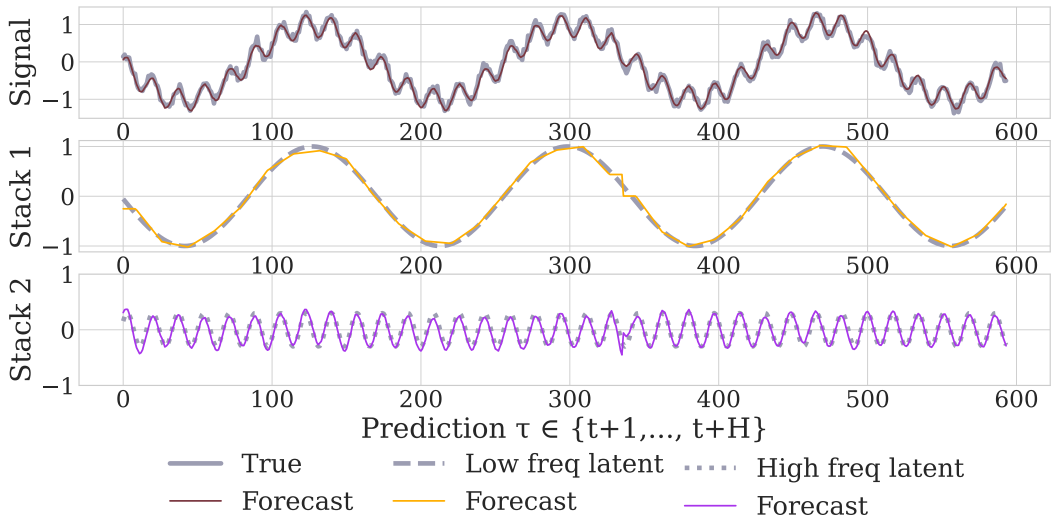
(iv) **Long Horizon Forecasting Comparison:** We achieve SoTA performance on six large-scale benchmark datasets: electricity transformer temperature, exchange rate, electricity consumption, San Francisco bay area highway traffic, weather and influenza-like illness.



(a) Computational Cost



(b) Prediction Errors



(c) Neural Hierarchical Interpolation

Figure 6.1: (a) The computational costs in time and memory (b) and *mean absolute errors* (MAE) of the predictions of a high capacity fully connected model exhibit evident deterioration with growing forecast horizons. (c) Specializing a flexible model's outputs in the different frequencies of the signal through hierarchical interpolation combined with multi-rate input processing offers a solution.

6.3 Related Work

Neural forecasting. Over the past few years, deep forecasting methods have become ubiquitous in industrial forecasting systems, with examples in optimal resource allocation and planning in transportation (Laptev et al., 2017), large e-commerce retail (Wen et al., 2017; Olivares et al., 2021; Paria et al., 2021; Rangapuram et al., 2021), or financial trading (Banushev and Barclay, 2021). The evident success of the methods in recent forecasting competitions (Makridakis et al., 2020a; Makridakis et al., 2021) has renovated the interest within the academic community (Benidis et al., 2020). In the context of multi-variate long-horizon forecasting, Transformer-based approaches have dominated the landscape in the recent years, including `Autoformer` (Wu et al., 2021), an encoder-decoder model with decomposition capabilities and an approximation to attention based on Fourier transform, `Informer` (Zhou et al., 2020), Transformer with MLP based multi-step prediction strategy, that approximates self-attention with sparsity, `Reformer` (Kitaev et al., 2020), Transformer that approximates attention with locality-sensitive hashing and `LogTrans` (Li et al., 2019), Transformer with local/log-sparse attention.

Multi-step forecasting. Investigations of the bias/variance trade-off in multi-step forecasting strategies reveal that the *direct* strategy, which allocates a different model for each step, has low bias and high variance, avoiding error accumulation across steps, exhibited by the classical *recursive* strategy, but losing in terms of net model parsimony. Conversely, in the *joint* forecasting strategy, a single model produces forecasts for all steps in one shot, striking the perfect balance between variance and bias, avoiding error accumulation and leveraging shared model parameters (Bao et al., 2014; Amir and Souhaib, 2016; Wen et al., 2017).

Multi-rate input sampling. Previous forecasting literature recognized challenges of extremely long horizon predictions, and proposed *mixed data sampling regression* (`MIDAS`) (Ghysels et al., 2007; Armesto et al., 2010) to ameliorate the problem of parameter proliferation while preserving high frequency temporal information. `MIDAS` regressions maintained the classic *recursive* forecasting strategy of linear auto-regressive models, but defined a parsimonious fashion of feeding the inputs.

Interpolation. Interpolation has been extensively used to augment the resolution of modeled signals in many fields such as signal and image processing (Meijering, 2002). In time-series forecasting, its applications range from completing unevenly sampled data and noise filters Chow and Lin, 1971; Fernandez, 1981; Shukla and Marlin, 2019; Rubanova et al., 2019 to fine-grained quantile-regressions with recurrent networks (Gasthaus et al., 2019). To our knowledge, temporal interpolation has not been used to induce multi-scale hierarchical time-series forecasts.

6.4 Methodology

In this section, we describe our proposed approach, NHITS, whose high-level diagram and main principles of operation are depicted in Figure 6.3. Our method extends the Neural Basis Expansion Analysis (NBEATS) approach (Oreshkin et al., 2020; Olivares et al., 2022a) in several important respects, making it more accurate and computationally efficient, especially in the context of long-horizon forecasting. In essence, our approach uses multi-rate sampling of the input signal and multi-scale synthesis of the forecast, resulting in a hierarchical construction of forecast, greatly reducing computational requirements and improving forecasting accuracy.

Similarly to NBEATS, NHITS performs local nonlinear projections onto basis functions across multiple blocks. Each block consists of a Multi Layer Perceptron (MLP), which learns to produce coefficients for the backcast and forecast outputs of its basis. The backcast output is used to clean the inputs of subsequent blocks, while the forecasts are summed to compose the final prediction. The blocks are grouped in stacks, each specialized in learning a different characteristic of the data using a different set of basis functions. The overall network input, $\mathbf{y}_{[t-L:t]}$, consists of L lags.

NHITS is composed of S stacks, B blocks each. Each block contains an MLP predicting forward and backward basis coefficients. The next subsections describe the novel components of our architecture. Note that in the following, we skip the stack index s for brevity.

6.4.1 Multi-Rate Signal Sampling

At the input to each block ℓ, we propose to use a MaxPool layer with kernel size k_ℓ to help it focus on analyzing components of its input with a specific scale. Larger k_ℓ will tend to cut more high-frequency/small-time-scale components from the input of the MLP, forcing the block to focus on analyzing large scale/low frequency content. We call this *multi-rate signal sampling*, referring to the fact that the MLP in each block faces a different effective input signal sampling rate. Intuitively, this helps the blocks with larger pooling kernel size k_ℓ focus on analyzing large scale components critical for producing consistent long-horizon forecasts.

Additionally, multi-rate processing reduces the width of the MLP input for most blocks, limiting the memory footprint and the amount of computation as well as reducing the number of learnable parameters and hence alleviating the effects of overfitting, while maintaining the original receptive field. Given block ℓ input $\mathbf{y}_{[t-L:t],\ell}$ (the input to the first block $\ell = 1$ is the network-wide input, $\mathbf{y}_{[t-L:t],1} \equiv \mathbf{y}_{[t-L:t]}$), this operation can be formalized as follows:

$$\mathbf{y}^{(p)}_{[t-L:t],\ell} = \mathbf{MaxPool}\left(\mathbf{y}_{[t-L:t],\ell},\ k_\ell\right) \tag{6.1}$$

6.4.2 Non-Linear Regression

Following subsampling, block ℓ looks at its input and non-linearly regresses forward θ^f_ℓ and backward θ^b_ℓ interpolation MLP coefficients that learns hidden vector $\mathbf{h}_\ell \in \mathbb{R}^{N_h}$, which is then linearly projected:

$$\begin{aligned}
\mathbf{h}_\ell &= \mathbf{MLP}_\ell\left(\mathbf{y}^{(p)}_{[t-L:t],\ell}\right) \\
\theta^f_\ell &= \mathbf{LINEAR}^f\left(\mathbf{h}_\ell\right) \\
\theta^b_\ell &= \mathbf{LINEAR}^b\left(\mathbf{h}_\ell\right)
\end{aligned} \tag{6.2}$$

The coefficients are then used to synthesize backcast $\tilde{\mathbf{y}}_{[t-L:t],\ell}$ and forecast $\hat{\mathbf{y}}_{[t+1:t+H],\ell}$ outputs of the block, via the process described below.

6.4.3 Hierarchical Interpolation

In most multi-horizon forecasting models, the cardinality of the neural network prediction equals the dimensionality of horizon, H. For example, in NBEATS-I $|\theta^f_\ell| = H$; in Transformer-based models, decoder attention layer cross-correlates H output embeddings with L encoded input embeddings (L tends to grow with growing H). This leads to quick inflation in compute requirements and unnecessary explosion in model expressiveness as horizon H increases.

To combat these issues, we propose to use *temporal interpolation*. We define the dimensionality of the interpolation coefficients in terms of the *expressiveness ratio* r_ℓ that controls the number of parameters per unit of output time, $|\theta^f_\ell| = \lceil r_\ell H \rceil$. To recover the original sampling rate and predict all H points in the horizon, we use temporal interpolation via the interpolation function g:

$$\begin{aligned}
\hat{y}_{\tau,\ell} &= g(\tau, \theta^f_\ell), \quad \forall \tau \in \{t+1, \ldots, t+H\}, \\
\tilde{y}_{\tau,\ell} &= g(\tau, \theta^b_\ell), \quad \forall \tau \in \{t-L, \ldots, t\}.
\end{aligned} \tag{6.3}$$

Interpolation can vary in *smoothness*, $g \in \mathcal{C}^0, \mathcal{C}^1, \mathcal{C}^2$. In Appendix G we explore the nearest neighbor, piece-wise linear and cubic alternatives. For concreteness, the linear interpolator $g \in \mathcal{C}^1$, along with the time partition $\mathcal{T} = \{t+1, t+1+1/r_\ell, \ldots, t+H-1/r_\ell, t+H\}$, is defined as

$$\begin{aligned}
g(\tau, \theta) &= \theta[t_1] + \left(\frac{\theta[t_2] - \theta[t_1]}{t_2 - t_1}\right)(\tau - t_1) \\
t_1 &= \arg\min_{t \in \mathcal{T}: t \leq \tau} \tau - t, \quad t_2 = t_1 + 1/r_\ell.
\end{aligned} \tag{6.4}$$

The *hierarchical* interpolation principle is implemented by distributing expressiveness ratios across blocks in a manner synchronized with multi-rate sampling. Blocks closer to the input

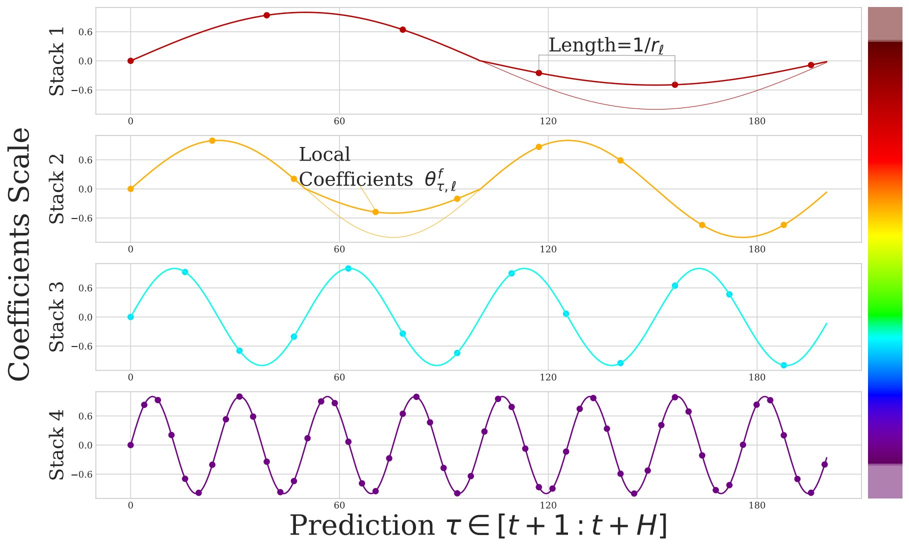


Figure 6.2: NHITS composes its predictions hierarchically using blocks specialized in different frequencies based on controlled signal projections, through *expressiveness ratios*, and interpolation of each block. The coefficients are locally determined along the horizon, allowing NHITS to reconstruct non-periodic/stationary signals, beyond constant Fourier transform projections.

have smaller r_ℓ and larger k_ℓ, implying that input blocks generate low-granularity signals via more aggressive interpolation, being also forced to look at more aggressively sub-sampled (and smoothed) signals. The resulting forecast $\hat{\mathbf{y}}_{[t+1:t+H]}$ is assembled by summing the outputs of all blocks, essentially composing it out of interpolations at different time-scale hierarchy levels.

Since each block specializes on its own scale of input and output signal, this induces a clearly structured hierarchy of interpolation granularity, the intuition conveyed in Figure 6.1 and Figure 6.2. We propose to use *exponentially increasing expressiveness ratios* to handle a wide range of frequency bands while controlling the number of parameters. Alternatively, each stack can specialize in modeling a different known cycle of the time-series (weekly, daily etc.) using a matching r_ℓ (see Table A.3). Finally, the backcast residual formed at previous hierarchy scale is subtracted from the input of the next hierarchy level to amplify the focus of the next level block on signals outside of the band that has already been handled by the previous hierarchy members.

$$\hat{\mathbf{y}}_{[t+1:t+H]} = \sum_{l=1}^{L} \hat{\mathbf{y}}_{[t+1:t+H],\ell}$$

$$\mathbf{y}_{[t-L:t],\ell+1} = \mathbf{y}_{[t-L:t],\ell} - \tilde{\mathbf{y}}_{[t-L:t],\ell}$$

Hierarchical interpolation has advantageous theoretical guarantees. We show in Appendix A, that it can approximate infinitely/dense horizons. As long as the interpolating function g is characterized by projections to informed multi-resolution functions V_w, and the forecast relationships are smooth.

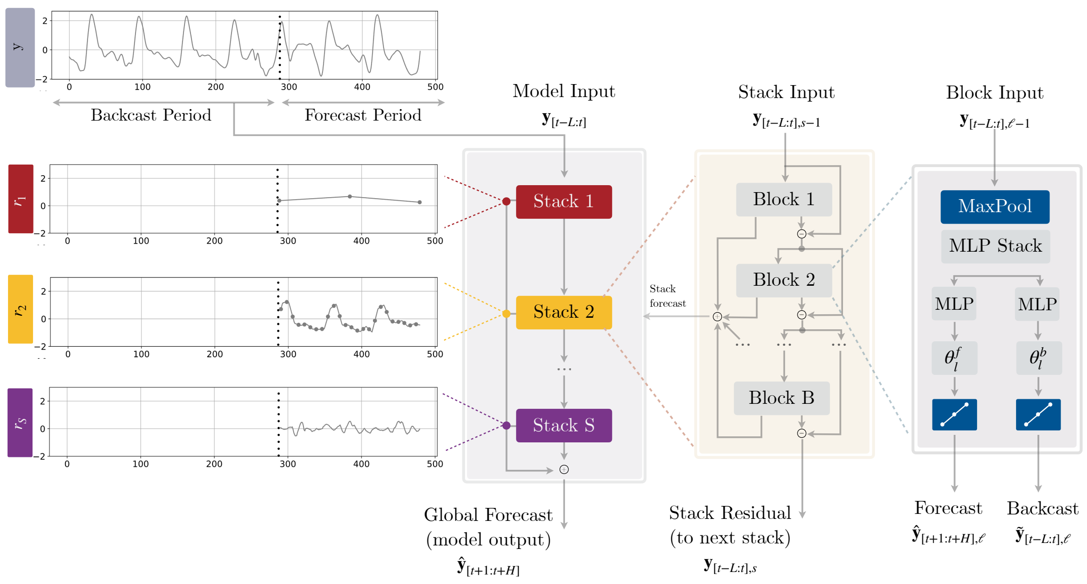


Figure 6.3: NHITS architecture. The model is composed of several MLPs with ReLU nonlinearities. Blocks are connected via doubly residual stacking principle with the backcast $\tilde{\mathbf{y}}_{t-L:t,\ell}$ and forecast $\hat{\mathbf{y}}_{t+1:t+H,\ell}$ outputs of the ℓ-th block. Multi-rate input pooling, hierarchical interpolation and backcast residual connections together induce the specialization of the additive predictions in different signal bands, reducing memory footprint and compute time, improving architecture parsimony and accuracy.

6.4.4 Neural Basis Approximation Theorem

The simplicity of NHITS provides allows to identify the sources of its performance improvements, which we back up with a strong theoretical foundation. NHITS' hierarchical interpolation extends the Fourier transform and connects the multi-step forecasting approach to non-linear wavelet projections (Daubechies, 1992), central to Shannon's interpolation theory.

Theorem 6.1: Neural Basis Approximation.

Let a forecast mapping be $\mathcal{Y}(\cdot \mid \mathbf{y}_{[t-L:t]}) : [0,1]^L \to \mathcal{F}$, where the forecast functions $\mathcal{F} = \{\mathcal{Y}(\tau) : [0,1] \to \mathbb{R}\} = \mathcal{L}^2([0,1])$ representing a infinite/dense horizon, are square integrable. If the multi-resolution functions $V_w = \{\phi_{w,h}(\tau) = \phi(2^w(\tau - h)) \mid w \in \mathbb{Z}, h \in 2^{-w} \times [0,\dots,2^w]\}$ can arbitrarily approximate $\mathcal{L}^2([0,1])$. And the projection $\mathrm{Proj}_{V_w}(\mathcal{Y}(\tau))$ varies smoothly on $\mathbf{y}_{[t-L:t]}$. Then the forecast mapping $\mathcal{Y}(\cdot \mid \mathbf{y}_{[t-L:t]})$ can be arbitrarily approximated by a neural basis expansion learning a finite number of multi-resolution coefficients $\hat{\theta}_{w,h}$. That is $\forall \epsilon > 0$,

$$\int |\mathcal{Y}(\tau \mid \mathbf{y}_{[t-L:t]}) - \sum_{w,h} \hat{\theta}_{w,h}(\mathbf{y}_{[t-L:t]})\phi_{w,h}(\tau)| d\tau \leq \epsilon \tag{6.5}$$

Examples of multi-resolution functions $V_w = \{\phi_{w,h}(\tau) = \phi(2^w(\tau - h)) \mid w \in \mathbb{Z}, h \in 2^{-w} \times [0,\dots,2^w]\}$ include piece-wise constants, piece-wise linear functions and splines with arbitrary approximation capabilities.

Table 6.1: Summary of datasets used in our empirical study. All the datasets are used in the multivariate forecasting experiments, univariate experiments are performed on $\mathtt{ETTm}_2$ and Exchange.

Dataset	Frequency	Time Series	Total Observations	Test Observations	Rolled forecast evaluation data points	Horizon (H)
$\mathtt{ETTm}_2$	15 Minute	7	403,200	80,640	5.81e7	$\{96, 192, 336, 720\}$
Exchange	Daily	8	60,704	12,136	8.74e6	$\{96, 192, 336, 720\}$
ECL	Hourly	321	8,443,584	1,688,460	1.22e9	$\{96, 192, 336, 720\}$
Traffic-L	Hourly	862	15,122,928	3,023,896	2.18e9	$\{96, 192, 336, 720\}$
Weather	10 Minute	21	1,106,595	221,319	1.59e8	$\{96, 192, 336, 720\}$
ILI	Weekly	7	6,762	1,351	9.73e5	$\{24, 36, 48, 60\}$

6.5 Experiments

In this section we present our empirical results, we follow the experimental settings from (Wu et al., 2021; Zhou et al., 2020) (NeurIPS 2021 and AAAI 2021 Best Paper Award). We first describe datasets, baselines and metrics used for the quantitative evaluation of our model. Table 6.3 presents our key results, demonstrating SoTA performance of our method relative to existing work. We then carefully describe the details of training and evaluation setups. We conclude the section by describing ablation studies.

6.5.1 Datasets

All large-scale datasets used in our empirical studies are publicly available and have been used in neural forecasting literature, particularly in the context of long-horizon (Lai et al., 2017; Zhou et al., 2019; Li et al., 2019; Wu et al., 2021). Table 6.1 summarizes their characteristics. Each set is normalized with the train data mean and standard deviation.

Electricity Transformer Temperature. The $\mathtt{ETTm}_2$ dataset measures an electricity transformer from a region of a province of China including oil temperature and variants of load (such as high useful load and high useless load) from July 2016 to July 2018 at a fifteen minutes frequency. **Exchange-Rate.** The Exchange dataset is a collection of daily exchange rates of eight countries relative to the US dollar. The countries include Australia, UK, Canada, Switzerland, China, Japan, New Zealand and Singapore from 1990 to 2016. **Electricity.** The ECL dataset reports the fifteen minute electricity consumption (KWh) of 321 customers from 2012 to 2014. For comparability, we aggregate it hourly. **San Francisco Bay Area Highway Traffic.** This Traffic-L dataset was collected by the California Department of Transportation, it reports road hourly occupancy rates of 862 sensors, from January 2015 to December 2016. **Weather.** This Weather dataset contains the 2020 year of 21 meteorological measurements recorded every 10 minutes from the Weather Station of the Max Planck Biogeochemistry Institute in Jena, Germany. **Influenza-like illness.** The ILI dataset reports weekly recorded influenza-like illness (ILI) patients from Centers for Disease Control and Prevention of the United States from 2002 to 2021. It is a ratio of ILI patients vs. the week's total.

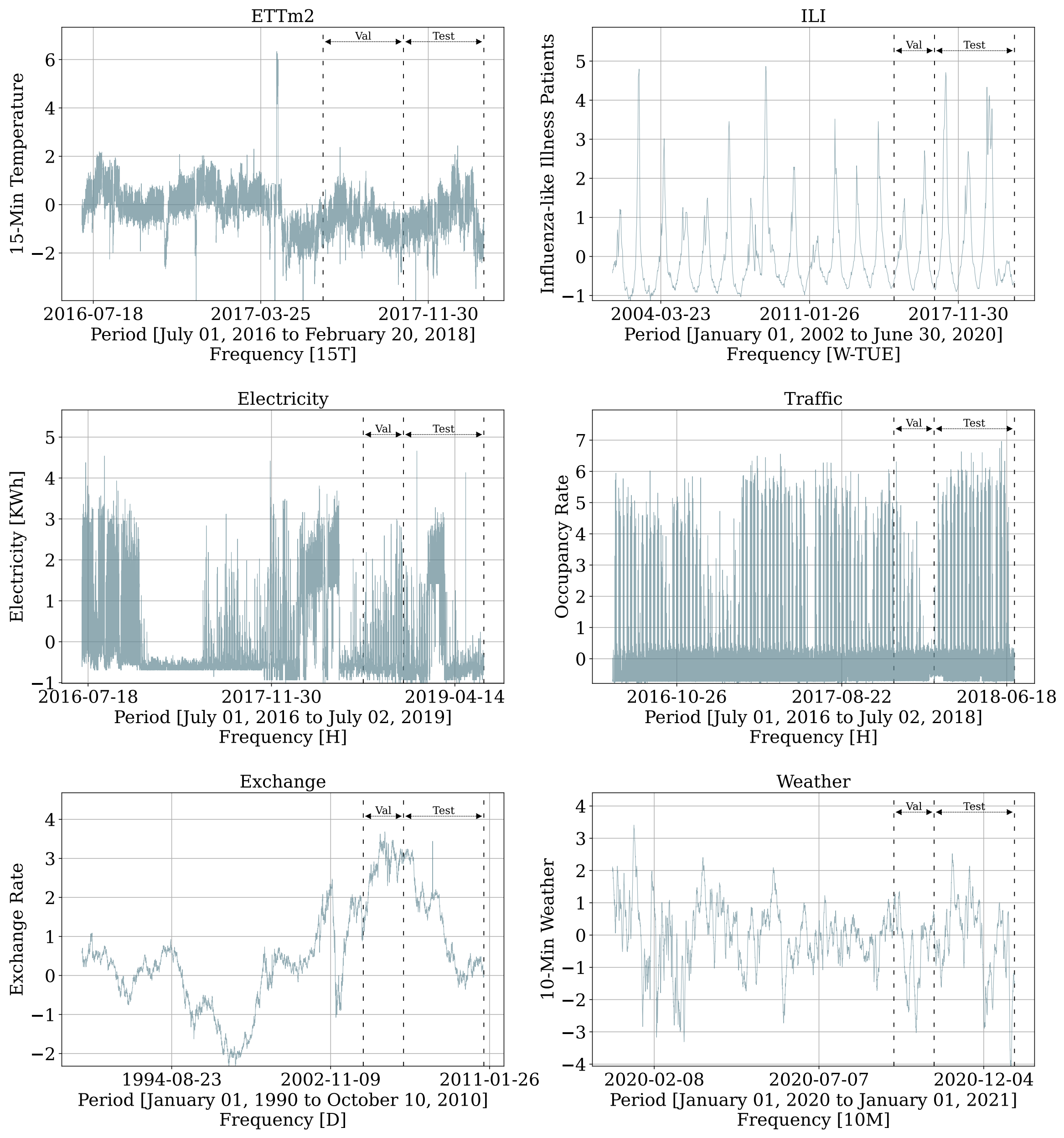


Figure 6.4: Datasets' partition into train, validation, and test sets used in our experiments (ETTm$_2$, ECL, `Exchange`, `ILI`, `Traffic-L`, and `Weather`). All use the last 20% of the total observations as test set (marked by the second dotted line), and the 10% preceding the test set as validation (between the first and second dotted lines), except for ETTm$_2$ that also use 20% as validation. Validation provides the signal for hyperparameter optimization. We construct test predictions using rolling windows.

Table 6.2: NHITS' explored hyperparameters.

Hyperparameter	Considered Values
Initial learning rate.	{1e-3}
Training steps.	{1000}
Random seed for initialization.	DiscreteRange(1, 10)
Input size multiplier (L=m*H).	$m \in \{5\}$
Batch Size.	{256}
Activation Function.	ReLU
Learning rate decay (3 times).	0.5
Pooling Kernel Size.	$[k_1, k_2, k_3] \in$ {[2,2,2], [4,4,4], [8,8,8], [8,4,1], [16,8,1]}
Number of Stacks.	$S \in \{3\}$
Number of Blocks in each stack.	$B \in \{1\}$
MLP Layers.	{2}
Coefficients Hidden Size.	$N_h \in \{512\}$
Number of Stacks' Coefficients.	$[r_1^{-1}, r_2^{-1}, r_3^{-1}] \in$ {[168,24,1], [24,12,1] [180,60,1], [40,20,1], [64,8,1] }
Interpolation strategy	$g(\tau, \theta) \in$ {Linear}

6.5.2 Evaluation

We evaluate the accuracy of our approach using Mean Absolute Error (MAE) and Mean Squared Error (MSE) metrics, which are well-established in the literature (Zhou et al., 2020; Wu et al., 2021), for varying horizon lengths H:

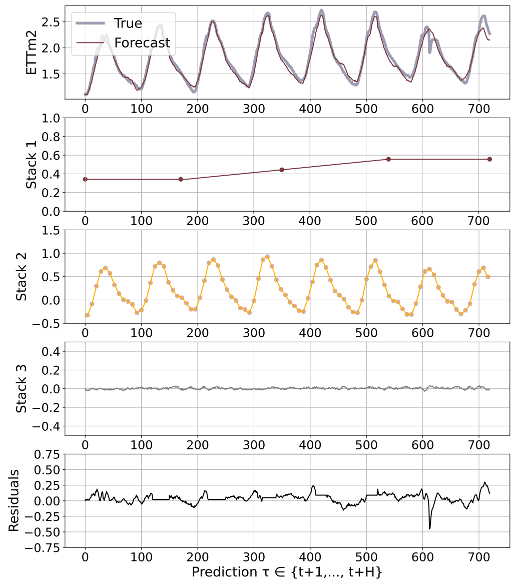
$$\mathrm{MSE} = \frac{1}{H}\sum_{\tau=t}^{t+H} \left(\mathbf{y}_{\tau} - \hat{\mathbf{y}}_{\tau}\right)^2, \qquad \mathrm{MAE} = \frac{1}{H}\sum_{\tau=t}^{t+H} \left|\mathbf{y}_{\tau} - \hat{\mathbf{y}}_{\tau}\right| \tag{6.6}$$

Note that for multivariate datasets, our algorithm produces forecast for each feature in the dataset and metrics are averaged across dataset features. Since our model is univariate, each variable is predicted using only its own history, $\mathbf{y}_{t-L:t}$, as input. Datasets are partitioned into train, validation and test splits. Train split is used to train model parameters, validation split is used to tune hyperparameters, and test split is used to compute metrics reported in Table 6.3.

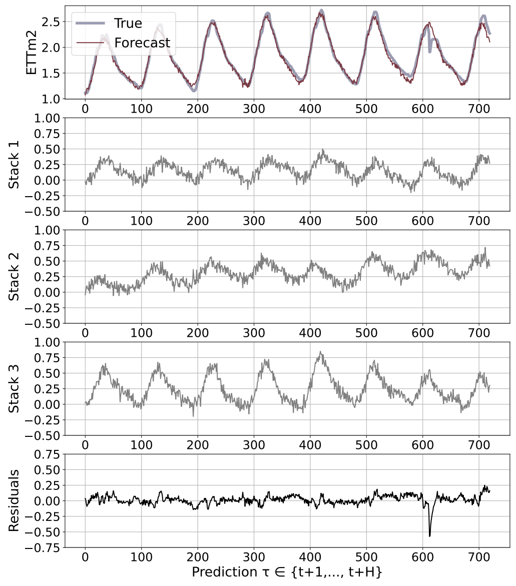
6.5.3 Train and Hyperparameter Optimization

We consider a minimal space of hyperparameters to explore configurations of the NHITS architecture. First, we consider the kernel pooling size for multi-rate sampling from Equation (6.1). Second, the number of coefficients from Equation (6.2) that we selected between several alternatives, some matching common seasonalities of the datasets and others exponentially increasing. We tune the random seed to escape underperforming local minima. Details in Appendix C.3's Table 6.2.

During the *hyperparameter optimization phase*, we measure MAE performance on the validation set and use a Bayesian optimization library (HYPEROPT; Bergstra et al. 2011), with 20 iterations. We use the optimal configuration based on the validation loss to make prediction on the test set. We refer to the hyperparameter optimization and test prediction as a *run*.



(a) H. interpolation, multi-rate sampling



(b) No h. interpolation, multi-rate sampling

Figure 6.5: ETTm2 and 720 ahead forecasts using NHITS (left panel), NHITS with linear interpolation and multi-rate sampling removed (right panel). The top row shows the original signal and the forecast. The second, third and fourth rows show the forecast components for each stack. The last row shows residuals, $y - \hat{y}$. In (a), each block shows scale specialization, unlike (b), in which signals are not interpretable.

NHITS is implemented in PyTorch (Paszke et al., 2019) and trained using Adaptive Moment Estimation SGD (ADAM) (Kingma and Ba, 2014), and MAE Empirical Risk Minimization (ERM), batch size 256 and initial learning rate of 1e-3, halved three times across the training procedure. All experiments use a GeForce RTX 2080 GPU.

6.5.4 Forecasting Results

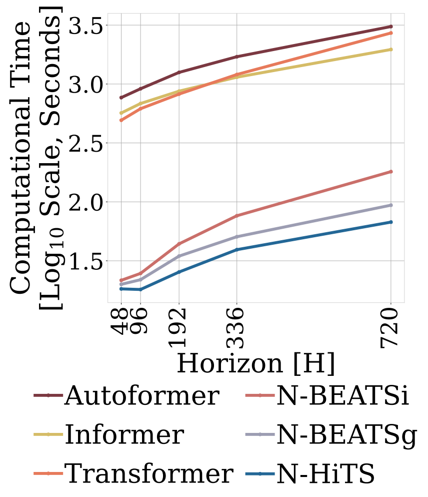
We compare NHITS to the following baselines: Transformer-based forecasting models (1) FEDformer (Zhou et al., 2022), (2) Autoformer (Wu et al., 2021), (3) Informer (Zhou et al., 2020), (4) Reformer (Kitaev et al., 2020) and (5) LogTrans (Li et al., 2019). Additionally, we consider the univariate baselines: (6) Dilated RNN (Chang et al., 2017) and (7) ARIMA (Hyndman and Khandakar, 2008).

Forecasting Accuracy. Table 6.3 summarizes the multivariate forecasting results. NHITS outperforms the best baseline, with average relative error decrease across datasets and horizons of 14% in MAE and 16% in MSE. NHITS maintains a comparable performance to other state-of-the-art methods for the shortest measured horizon (96/24), while for the longest measured horizon (720/60) decreases multivariate MAE by 11% and MSE by 17%. We complement the key results in Table 6.3, with the additional univariate forecasting experiments in Appendix F, again demonstrating state-of-the-art performance against baselines.

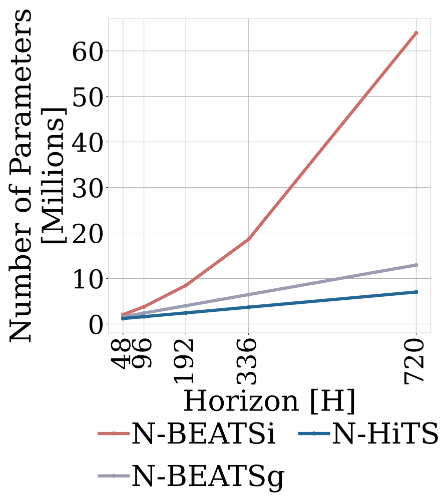
Table 6.3: Key empirical results in long-horizon forecasting setup, lower scores are better. Metrics are averaged over eight runs and standard deviation in brackets, best results are highlighted in bold, second best results are highlighted in blue.

* Caveats of the concurrent research comparison are that these articles have not yet been peer-reviewed; some evaluations deviate in the length of the forecast horizons or don't report results for all the benchmark datasets, and finally, unfortunately, most studies only report a single run or don't report standard deviations in their accuracy measurements for which it is difficult to assess the significance of the results.

		NHITS (Ours)		Autoformer		Informer		LogTrans		Reformer		ARIMA		FEDformer		ETSformer*		Preformer*	
	Horizon	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm$_2$	96	**0.176** (0.003)	**0.255** (0.001)	0.255 (0.020)	0.339 (0.020)	0.365 (0.062)	0.453 (0.047)	0.768 (0.071)	0.642 (0.020)	0.658 (0.121)	0.619 (0.021)	0.225 (-)	0.301 (-)	0.203 (-)	0.287 (-)	0.183 (-)	0.275 (-)	0.213 (-)	0.295 (-)
	192	**0.245** (0.005)	**0.305** (0.002)	0.281 (0.027)	0.340 (0.025)	0.533 (0.109)	0.563 (0.050)	0.989 (0.124)	0.757 (0.049)	1.078 (0.106)	0.827 (0.012)	0.298 (-)	0.345 (-)	0.269 (-)	0.328 (-)	- (-)	- (-)	0.269 (-)	0.329 (-)
	336	**0.295** (0.004)	**0.346** (0.002)	0.339 (0.018)	0.372 (0.015)	1.363 (0.173)	0.887 (0.056)	1.334 (0.168)	0.872 (0.054)	1.549 (0.146)	0.972 (0.0)	0.370 (-)	0.386 (-)	0.325 (-)	0.366 (-)	- (-)	- (-)	0.324 (-)	0.363 (-)
	720	**0.401** (0.013)	**0.413** (0.009)	0.422 (0.015)	0.419 (0.010)	3.379 (0.143)	1.388 (0.037)	3.048 (0.140)	1.328 (0.023)	2.631 (0.126)	1.242 (0.014)	0.478 (-)	0.445 (-)	0.421 (-)	0.415 (-)	- (-)	- (-)	0.418 (-)	0.416 (-)
ECL	96	**0.147** (0.002)	**0.249** (0.002)	0.201 (0.003)	0.317 (0.004)	0.274 (0.004)	0.368 (0.003)	0.258 (0.002)	0.357 (0.002)	0.312 (0.003)	0.402 (0.004)	1.220 (-)	0.814 (-)	0.183 (-)	0.297 (-)	0.187 (-)	0.302 (-)	0.180 (-)	0.297 (-)
	192	**0.167** (0.005)	**0.269** (0.005)	0.222 (0.003)	0.334 (0.004)	0.296 (0.009)	0.386 (0.007)	0.266 (0.005)	0.368 (0.004)	0.348 (0.004)	0.433 (0.005)	1.264 (-)	0.842 (-)	0.195 (-)	0.308 (-)	0.196 (-)	0.311 (-)	0.189 (-)	0.302 (-)
	336	**0.186** (0.001)	**0.290** (0.001)	0.231 (0.006)	0.338 (0.004)	0.300 (0.007)	0.394 (0.004)	0.280 (0.006)	0.380 (0.001)	0.350 (0.004)	0.433 (0.003)	1.311 (-)	0.866 (-)	0.212 (-)	0.313 (-)	0.215 (-)	0.330 (-)	0.201 (-)	0.319 (-)
	720	0.243 (0.008)	**0.340** (0.007)	0.254 (0.007)	0.361 (0.008)	0.373 (0.034)	0.439 (0.024)	0.283 (0.003)	0.376 (0.002)	0.340 (0.002)	0.42 (0.002)	1.364 (-)	0.891 (-)	**0.231** (-)	0.343 (-)	0.236 (-)	0.348 (-)	0.232 (-)	0.342 (-)
Exchange	96	0.092 (0.002)	0.02 (0.002)	0.197 (0.019)	0.323 (0.012)	0.847 (0.150)	0.752 (0.060)	0.968 (0.177)	0.812 (0.027)	1.065 (0.070)	0.829 (0.013)	0.296 (-)	0.214 (-)	0.139 (-)	0.276 (-)	**0.083** (-)	**0.202** (-)	0.148 (-)	0.282 (-)
	192	0.208 (0.025)	0.322 (0.020)	0.300 (0.020)	0.369 (0.016)	1.204 (0.149)	0.895 (0.061)	1.040 (0.232)	0.851 (0.029)	1.188 (0.041)	0.906 (0.008)	1.056 (-)	0.326 (-)	0.256 (-)	0.369 (-)	**0.180** (-)	**0.302** (-)	0.268 (-)	0.378 (-)
	336	**0.301** (0.042)	**0.403** (0.030)	0.509 (0.041)	0.524 (0.016)	1.672 (0.036)	1.036 (0.014)	1.659 (0.122)	1.081 (0.015)	1.357 (0.027)	0.976 (0.010)	2.298 (-)	0.467 (-)	0.426 (-)	0.464 (-)	0.354 (-)	0.433 (-)	0.447 (-)	0.499 (-)
	720	**0.798** (0.041)	**0.596** (0.013)	1.447. (0.084)	0.941 (0.028)	2.478 (0.198)	1.310 (0.070)	1.941 (0.327)	1.127 (0.030)	1.510 (0.071)	1.016 (0.008)	20.666 (-)	0.864 (-)	1.090 (-)	0.800 (-)	0.996 (-)	0.761 (-)	1.092 (-)	0.812 (-)
Traffic-L	96	**0.402** (0.005)	**0.282** (0.002)	0.613 (0.028)	0.388 (0.012)	0.719 (0.150)	0.391 (0.060)	0.684 (0.177)	0.384 (0.027)	0.732 (0.070)	0.423 (0.013)	1.997 (-)	0.924 (-)	0.562 (-)	0.349 (-)	0.614 (-)	0.395 (-)	0.560 (-)	0.349 (-)
	192	**0.420** (0.002)	**0.297** (0.003)	0.616 (0.042)	0.382 (0.020)	0.696 (0.050)	0.379 (0.023)	0.685 (0.055)	0.39 (0.021)	0.733 (0.013)	0.42 (0.011)	2.044 (-)	0.944 (-)	0.562 (-)	0.346 (-)	0.629 (-)	0.398 (-)	0.565 (-)	0.349 (-)
	336	**0.448** (0.006)	**0.313** (0.003)	0.622 (0.009)	0.337 (0.003)	0.777 (0.069)	0.420 (0.026)	0.733 (0.012)	0.408 (0.008)	0.742 (0.0)	0.42 (0.0)	2.096 (-)	0.960 (-)	0.570 (-)	0.323 (-)	0.646 (-)	0.417 (-)	0.577 (-)	0.351 (-)
	720	**0.539** (0.022)	**0.353** (0.012)	0.660 (0.025)	0.408 (0.015)	0.864 (0.026)	0.472 (0.015)	0.717 (0.030)	0.396 (0.010)	0.755 (0.023)	0.423 (0.014)	2.138 (-)	0.971 (-)	0.596 (-)	0.368 (-)	0.631 (-)	0.389 (-)	0.597 (-)	0.358 (-)
Weather	96	**0.158** (0.002)	**0.195** (0.002)	0.266 (0.007)	0.336 (0.006)	0.300 (0.013)	0.384 (0.013)	0.458 (0.143)	0.49 (0.038)	0.689 (0.042)	0.596 (0.019)	0.217 (-)	0.258 (-)	0.217 (-)	0.296 (-)	0.189 (-)	0.272 (-)	0.227 (-)	0.292 (-)
	192	**0.211** (0.001)	**0.247** (0.003)	0.307 (0.024)	0.367 (0.022)	0.598 (0.045)	0.544 (0.028)	0.658 (0.151)	0.589 (0.032)	0.752 (0.048)	0.638 (0.029)	0.263 (-)	0.299 (-)	0.276 (-)	0.336 (-)	0.231 (-)	0.303 (-)	0.275 (-)	0.322 (-)
	336	**0.274** (0.009)	**0.300** (0.008)	0.359 (0.035)	0.395 (0.031)	0.578 (0.024)	0.523 (0.016)	0.797 (0.034)	0.652 (0.019)	0.064 (0.030)	0.596 (0.021)	0.330 (-)	0.347 (-)	0.339 (-)	0.380 (-)	0.305 (-)	0.357 (-)	0.324 (-)	0.352 (-)
	720	**0.351** (0.020)	**0.353** (0.016)	0.419 (0.017)	0.428 (0.014)	1.059 (0.096)	0.741 (0.042)	0.869 (0.045)	0.675 (0.093)	1.130 (0.084)	0.792 (0.055)	0.425 (-)	0.405 (-)	0.403 (-)	0.428 (-)	0.352 (-)	0.391 (-)	0.394 (-)	0.393 (-)
ILI	24	**1.862** (0.064)	**0.869** (0.020)	3.483 (0.107)	1.287 (0.018)	5.764 (0.354)	1.677 (0.080)	4.480 (0.313)	1.444 (0.033)	4.4 (0.177)	1.382 (0.021)	5.554 (-)	1.434 (-)	2.203 (-)	0.963 (-)	2.862 (-)	1.128 (-)	3.143 (-)	1.185 (-)
	36	**2.071** (0.015)	**0.934** (0.003)	3.103 (0.139)	1.148 (0.025)	4.755 (0.248)	1.467 (0.067)	4.799 (0.251)	1.467 (0.023)	4.783 (0.138)	1.448 (0.023)	6.940 (-)	1.676 (-)	2.272 (-)	0.976 (-)	2.683 (-)	1.029 (-)	2.793 (-)	1.054 (-)
	48	**2.134** (0.142)	**0.932** (0.034)	2.669 (0.151)	1.085 (0.037)	4.763 (0.295)	1.469 (0.059)	4.800 (0.233)	1.468 (0.021)	4.832 (0.122)	1.465 (0.016)	7.192 (-)	1.736 (-)	2.209 (-)	0.981 (-)	2.456 (-)	0.986 (-)	2.845 (-)	1.090 (-)
	60	**2.137** (0.075)	**0.968** (0.012)	2.770 (0.085)	1.125 (0.019)	5.264 (0.237)	1.564 (0.044)	5.278 (0.231)	1.56 (0.014)	4.882 (0.123)	1.483 (0.016)	6.648 (-)	1.656 (-)	2.545 (-)	1.061 (-)	2.630 (-)	1.057 (-)	2.957 (-)	1.124 (-)



(a) NHITS-NaiveBU



(b) NHITS-GroupBU

Figure 6.6: Computational efficiency comparison. NHITS exhibits the best training time compared to Transformer-based and fully connected models, and smallest memory footprint.

Computational Efficiency. We measure the computational training time of NHITS, NBEATS and Transformer-based methods in the multivariate setting and show compare in Figure 6.6. The experiment monitors the whole training process for the ETTm_2 dataset. For the Transformer-based models we used hyperparameters reported in (Wu et al., 2021). Compared to the Transformer-based methods, NHITS is $45\times$ faster than Autoformer. In terms of memory, NHITS has less than 26% of the parameters of the second-best alternative, since it scales linearly with respect to the input's length. Compared to the original NBEATS, our method is $1.26\times$ faster and requires only 54% of the parameters. Finally, while NHITS is an univariate model, it has *global* (shared) parameters for all time-series in the dataset.

6.5.5 Ablation Studies

We believe that the advantages of the NHITS architecture are rooted in its multi-rate hierarchical nature. Figure 6.5 shows a qualitative comparison of NHITS with and without hierarchical interpolation/multi-rate sampling components. We clearly see NHITS developing the ability to produce interpretable forecast decomposition providing valuable information about trends and seasonality in separate channels, unlike the control model. Appendix G presents the decomposition for the different interpolation techniques. We support our qualitative conclusion with quantitative results. We define the following set of alternative models: NHITS, our proposed model with both multi-rate sampling and hierarchical interpolation, NHITS_2 only hierarchical interpolation, NHITS_3 only multi-rate sampling, NHITS_4 no multi-rate sampling or interpolation (corresponds to the original NBEATS-G (Oreshkin et al., 2020)), finally NBEATS-I, the interpreatble version of the NBEATS. Table 6.4 clearly shows that the combination of both proposed components (hierarchical interpolation and multi-rate sampling) results in the best performance, emphasizing their complementary nature in long-horizon forecasting. We see that the original NBEATS is consistently worse, especially the NBEATS-I. The advantages of the proposed techniques for long-horizon forecasting, multi-rate sampling and interpolation, are not limited to the NHITS architecture. In Appendix H we demonstrate how adding them to a DilRNN improve its performance.

Table 6.4: Empirical evaluation of long multi-horizon multivariate forecasts for NHITS with/without enhancements. MAE and MSE for predictions averaged over eight runs, and five datasets, the best result is highlighted in bold, second best in blue (lower is better).

		NHITS	$NHITS_2$	$NHITS_3$	$NHITS_4$	NBEATS-I
A. MSE	96	0.195	0.196	**0.192**	0.196	0.209
	192	**0.250**	0.261	0.251	0.263	0.266
	336	**0.315**	0.315	0.342	0.346	0.408
	720	**0.484**	0.498	0.518	0.548	0.794
A. MAE	96	0.239	0.241	**0.237**	0.240	0.254
	192	**0.290**	0.299	0.291	0.300	0.307
	336	**0.338**	0.342	0.346	0.352	0.405
	720	**0.439**	0.450	0.454	0.468	0.597

Additional *ablation studies* are reported in Appendix G. The MaxPool multi-rate sampling wins over AveragePool. Linear interpolation wins over nearest neighbor and cubic. Finally and most importantly, we show that the order in which hierarchical interpolation is implemented matters significantly. The best configuration is to have the low-frequency/large-scale components synthesized and removed from analysis first, followed by more fine-grained modeling of high-frequency/intermittent signals.

6.6 Conclusion

Our results indicate the complementarity and effectiveness of multi-rate sampling and hierarchical interpolation for long-horizon time-series forecasting. Table 6.4 indicates that these components enforce a useful inductive bias compared to both the free-form model $NHITS_4$ (plain fully connected architecture) and the parametric model NBEATS-I (polynomial trend and sinusoidal seasonality used as basis functions in two respective stacks). The latter obviously providing a detrimental inductive bias for long-horizon forecasting. Notwithstanding our current success, we believe we barely scratched the surface in the right direction and further progress is possible using advanced multi-scale processing approaches in the context of time-series forecasting, motivating further research.

NHITS outperforms SoTA baselines while simultaneously providing an interpretable non-linear decomposition. Figure 6.1 and Figure 6.5 showcase NHITS perfectly specializing and reconstructing latent harmonic signals from synthetic and real data respectively. This novel *interpretable* decomposition can provide insights to users, improving their confidence in high-stakes applications like healthcare. Finally, NHITS hierarchical interpolation can be explored from the multi-resolution analysis perspective (Daubechies, 1992). Replacing the sequential projections from the interpolation functions onto these Wavelet induced spaces is an interesting line of research.

Our study raises a question about the effectiveness of the existing long-horizon multi-variate forecasting approaches, as all of them are substantially outperformed by our univariate algorithm. If these approaches underperform due to problems with overfitting and model parsimony at the level of marginals, it is likely that the integration of our approach with Transformer-inspired architectures could form a promising research direction as the univariate results in Appendix C.4 suggest. However, there is also a chance that the existing approaches underperform due to their inability to effectively integrate information from multiple variables, which clearly hints at possibly untapped research potential in this area. Whichever is the case, we believe our results provide a strong guidance signal and a valuable baseline for future research in the area of long-horizon multi-variate forecasting.

We proposed a novel neural forecasting algorithm `NHITS` that combines two complementary techniques, multi-rate input sampling and hierarchical interpolation, to produce drastically improved, interpretable and computationally efficient long-horizon time-series predictions. Our model, operating in the univariate regime and accepting only the predicted time-series' history, significantly outperforms all previous Transformer-based multi-variate models using an order of magnitude less computation. This sets a new baseline for all ensuing multi-variate work on six popular datasets and motivates further research to effectively use information from multiple variables.

Part III

Conclusion

It’s a dangerous business, going out your door.
You step onto the road, and if you don’t keep your feet, there’s no knowing where you might be swept off to.

J. R. R. Tolkien

Chapter 7

Main Results and Future Work

There is often an imagined divide between statistical and econometric forecasting models and the machine learning approach. Statistical models are usually built on theoretically-inspired assumptions about the relationships between variables. Meanwhile, machine learning forecasting models are highly flexible and make predictions based on data without explicit programming, making minimal assumptions. Yet plenty of opportunities exist to improve forecasting methods through cross-pollination between statistics, econometrics, and machine learning.

We started the work with a simple question:

> *Can econometrics and statistical innovations be combined to advance usability, usefulness, and real-world impact of machine learning-based forecasting?*

We sought to rediscover the connection between machine learning-based forecasting with classic approaches and find a new balance between their extremes. This middle ground would enable us to balance the excess and deficiency in the forecasting methods' flexibility and rigid assumptions. Incorporating statistical and econometric approaches can effectively act as a form of regularization that balances the complexity of the model with its ability to generalize to new data while simultaneously enabling human understanding of its predictions and improving its computational efficiency. To test these ideas, we organized the thesis around the following hypothesis:

Thesis statement.

> *Confining machine learning-based forecasting methods with econometric and statistical domain knowledge is necessary to improve their accuracy, interpretability, and efficiency.*

We presented and investigated three neural forecasting methods that enhance state-of-the-art deep learning architectures with statistical and econometric intuitions. This final chapter summarizes our findings and their relevance to our hypothesis. Our experiments have shown promising results, matching our expectations, with evidence of improved interpretability, accuracy, and computational efficiency of our novel forecasting methods. These findings represent an important step toward multiple avenues of future research that we discuss in the last section.

7.1 Main Results

In Part I of our thesis, we provided an introduction to the forecasting task and its principles, along with a look into the modern approaches in the field. Chapter 2 covered the basics of the regression problem, forecast evaluation methods, common model estimation, and optimization techniques. In Chapter 2, we provided an overview of various forecast modeling approaches, including statistical, econometric, and machine learning techniques, which served as a reference for the thesis work.

In Part II, we presented the thesis's main contributions through four case studies that combined neural forecasting methods with econometric and statistical inspirations. Chapter 4 introduced NBEATSx, which extended neural basis expansion analysis with exogenous variables. In Chapter 5, we presented a novel probabilistic mixture model that can effectively tackle the hierarchical forecasting task. This new method extended the network's capabilities to approximate distributions, including those with coherence constraints. Chapter 6 introduced NHITS, a Wavelet analysis-inspired approach that specializes in multi-step forecasting strategies across different frequencies through time, addressing the long-horizon forecasting challenge.

The results presented in Part II proved our approach is a promising direction to enhance and guide the design of neural forecasting algorithms. By leveraging the strengths of econometric, statistical, and machine learning-based methods, two new methods introduced signal decomposition techniques capable of extracting insights and human understanding of the methods. All the techniques showed improved accuracy compared to the state-of-the-art while significantly reducing the computational costs of the methods.

7.2 Future Work

In this section we briefly introduce some of the most interesting lines of work extending on the ideas presented in the thesis. These ideas are a complement to those included in the conclusion of each case study from Part II. We believe that some of these ideas have the potential of defining the future of the forecasting task.

Transfer learning is one of the most outstanding achievements in machine learning and has many practical applications, yet for time series forecasting, the technique needs to be explored more. Training large forecasting models on vast data will continue to improve prediction accuracy. Still, more importantly, it will bypass the trade-off between accuracy and speed and enable time series forecasting without the need of a specific time series past information.

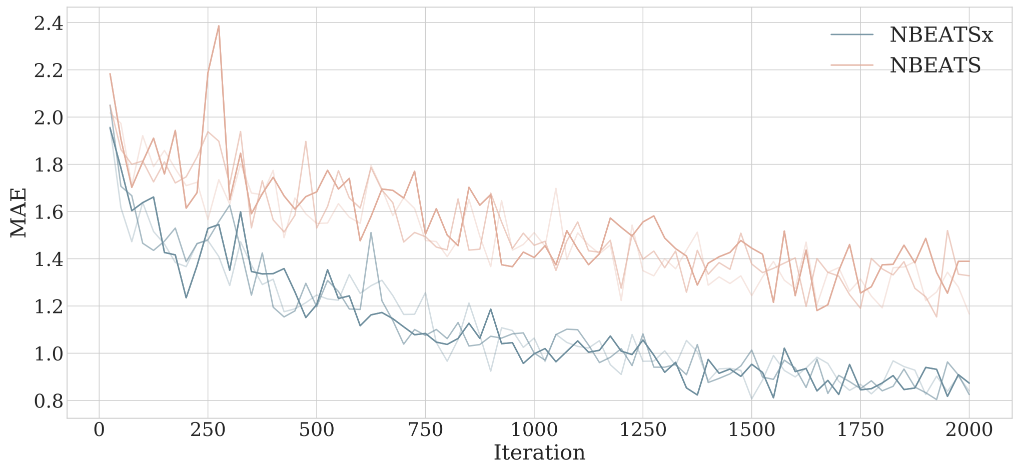
Forecasting aims to predict the future as accurately as possible, conditional on all the available information. Planning, on the other side, involves defining appropriate actions based on goals and forecasts; by its nature, planning is intended to lead to a specific course of action. An exciting research avenue is to develop algorithms that shorten the distance between the predictions and the actions. Training algorithms to take actions directly will see them shaping the future instead of only describing it.

Appendices

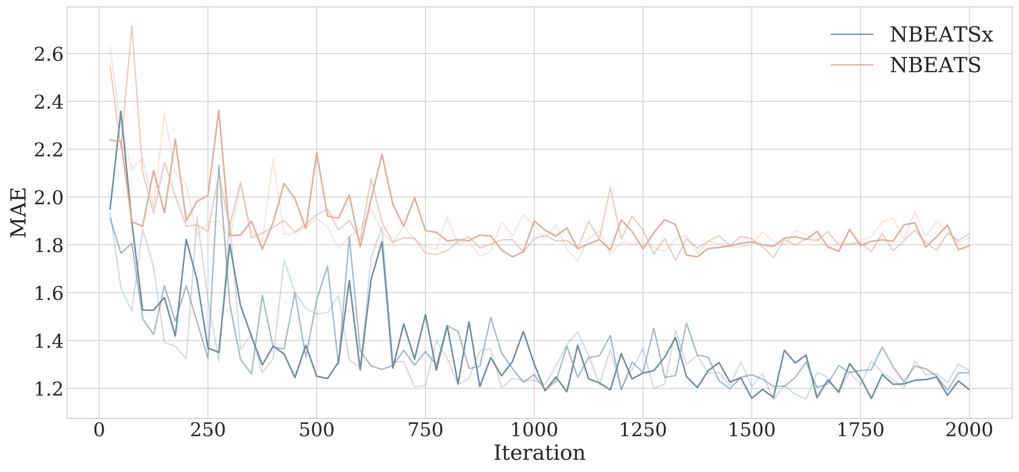
Appendix A

Interpretable Neural Forecasting

A.1 Training and validation curves



(a) Train Set



(b) Validation Set

Figure A.1: Training and validation *Mean Absolute Error* (MAE) curves on the NP market. We show the curves for NBEATSx-G with exogenous variables and NBEATS without exogenous variables as a function of optimization iterations. We define four curves by a different random seed used for initialization.

To study exogenous variables effects on the NBEATS model, we performed model training procedure diagnostics. Figure A.1 shows the train and validation *mean absolute error* (MAE) for the NBEATS and NBEATSx models as training progresses. The curves correspond to the hyperparameter optimization phase described in Section 4.5.4. The models trained with and without exogenous variables display a considerable difference in their train and validation errors as observed by the two separate clusters of trajectories. The exogenous variables, in this case, the electricity load and production forecasts, improve the neural basis expansion analysis.

A.1.1 Computational Time

Table A.1: Computational time performance in seconds for the top four most accurate models for the day-ahead electricity price forecasting task in the NP market, averaged for the four elements of the ensembles (Time performance for the rest of the markets is almost identical).

	LEARx	DNN	NBEATSx-G	NBEATSx-I
Recalibration	18.57	50.65	75.02	81.61
Prediction	0.0032	0.0041	0.0048	0.0054

We measured the computational time of the top four best algorithms with two metrics: the recalibration of the ensemble models selected from the hyperparameter optimization, and the computation of the predictions. For these experiments, we used a GeForce RTX 2080 GPU for the neural network models and an Intel(R) Xeon(R) Silver 4210 CPU @ 2.20GHz for LEAR.

The time of the *recalibration phase* of NBEATSx remains efficient, as it still trains in 75 and 81 seconds, increasing by 30 seconds on the DNN. The computational time of the prediction remains within miliseconds. Finally the *hyperparameter optimization* scales linearly with respect to the time of the *recalibration phase* and the evaluation steps of the optimization, in case of the NBEATSx-G the approximate time of a hyperparameter search of 1000 steps takes two days[1].

[1]For comparability we use 1000 steps (Lago et al., 2021a), restricting to 300 steps yields similar results.

Table A.2: Forecast accuracy measures for day-ahead electricity prices for the best *single model* out of the four models described in the Section 4.5.5. The ESRNN and NBEATS, are the original implementations and do not include time dependent covariates. The reported metrics are *mean absolute error* (MAE), *relative mean absolute error* (rMAE), *symmetric mean absolute percentage error* (sMAPE) and *root mean squared error* (RMSE). The smallest errors in each row are highlighted in bold.
[*] The EPEX-DE's LEARxresults differ from Lago et al., 2021a – the values are revised (Lago et al., 2021b)

		AR1	ESRNN	NBEATS	ARx1	LEARx[*]	DNN	NBEATSx-G	NBEATSx-I
NP	MAE	2.28	2.11	2.11	2.11	1.95	1.71	**1.65**	1.68
	rMAE	0.72	0.67	0.67	0.67	0.62	0.54	**0.52**	0.53
	sMAPE	6.51	6.09	6.06	6.1	5.62	4.97	**4.83**	4.89
	RMSE	4.08	3.92	3.98	3.84	3.60	3.36	**3.27**	3.33
PJM	MAE	3.88	3.63	3.48	3.68	3.09	3.07	3.02	**3.01**
	rMAE	0.8	0.75	0.72	0.76	0.64	0.63	**0.62**	**0.62**
	sMAPE	14.66	14.26	13.56	14.09	12.54	12.00	11.97	**11.91**
	RMSE	6.26	5.87	5.59	5.94	5.14	5.20	5.06	**5.00**
EPEX-BE	MAE	7.04	7.01	6.83	7.05	6.59	**6.07**	6.14	6.17
	rMAE	0.86	0.86	0.83	0.86	0.80	**0.74**	0.75	0.75
	sMAPE	16.29	15.95	16.03	16.21	15.95	**14.11**	14.68	14.52
	RMSE	17.25	16.76	16.99	17.07	16.29	15.95	15.46	**15.43**
EPEX-FR	MAE	4.74	4.68	4.79	4.85	4.25	4.06	3.98	**3.97**
	rMAE	0.80	0.78	0.80	0.86	0.71	0.68	**0.67**	**0.67**
	sMAPE	13.49	13.25	13.62	16.21	13.25	11.49	**11.07**	11.29
	RMSE	13.68	11.89	12.09	17.07	**10.75**	11.77	11.61	11.08
EPEX-DE	MAE	5.73	5.64	5.37	4.58	3.93	3.59	3.46	**3.37**
	rMAE	0.71	0.70	0.67	0.57	0.49	0.45	0.43	**0.42**
	sMAPE	21.22	21.09	19.71	18.52	16.80	14.68	14.78	**14.34**
	RMSE	9.39	9.17	9.03	7.69	6.53	6.08	5.84	**5.64**

A.2 Best Single Models

Table A.2 shows that the best NBEATSx models yield improvements of 14.8% on average across all the evaluation metrics when compared to its NBEATS counterpart without exogenous covariates, and improvements of 23.9% when compared to ESRNN without time-dependent covariates. A perhaps more remarkable result is the statistically significant improvement of forecast accuracy over LEAR and DNN benchmarks, ranging from 0.75% to 7.2% across all metrics and markets, with the exception of EPEX-BE. Compared to DNN, the RMSE improved on average 4.9%, the MAE improved 3.2%, the rMAE improved 3.0%, and sMAPE improved 1.7%. When comparing the best NBEATSx models against the best DNN on individual markets, NBEATSx improved by 3.18% on the Nord Pool market (NP), 2.03% 2.65% on French (EPEX-FR) and 5.24% on German (EPEX-DE) power markets. The positive difference in performance for Belgian (EPEX-BE) market of 0.53% was not statistically significant.

Figure A.2 provides a graphical representation of the GW test for the six best models, across the five markets for the MAE evaluation metric. The models included in the significance tests are the same as in Table A.2: LEAR, DNN, the ESRNN, NBEATS, and our proposed methods, the NBEATSx-G and NBEATSx-I. The p-value of each individual comparison shows if the improvement in performance (measured by MAE or RMSE) of the x-axis model over the y-axis model is statistically significant. Both the NBEATSx-G and NBEATSx-I model outperformed the LEAR and DNN models in all markets, with the exception of Belgium. Moreover, no benchmark outperformed the NBEATSx-I and NBEATSx-G on any market.

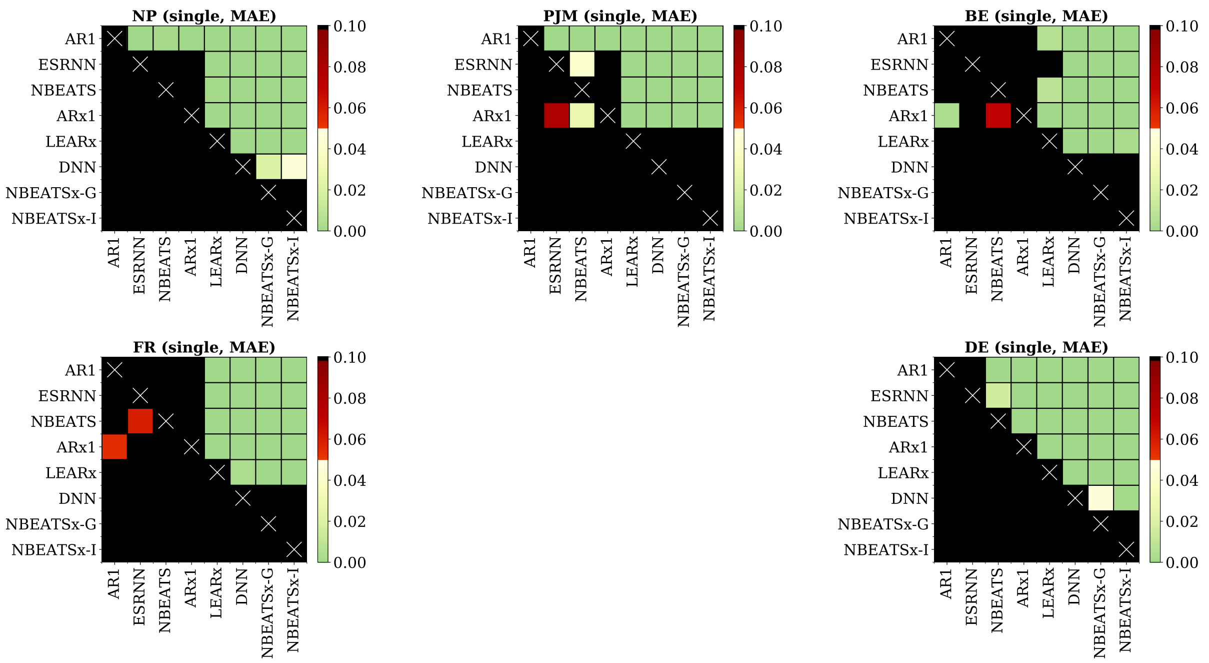


Figure A.2: Giacomini-White test for the day-ahead predictions with *mean absolute error* (MAE) applied to single model pairs on the five electricity markets datasets. Each grid represents one market. Each colored cell in a grid is plotted black, unless the predictions of the model corresponding to its column of the grid outperforms the predictions of the model corresponding to its row of the grid. The color scale reflects significance of the difference in MAE, with solid green representing the lowest p-values.

Appendix B

Probabilistic Hierarchical Forecasting

B.1 HINT's Probabilistic Coherence

In this Appendix we prove that HINT's probabilistic coherence. Given access to the joint bottom-level forecast probability $\hat{\mathbb{P}}_{[b]}$, and the *aggregation rule* for $\hat{\mathbb{P}}_{[a]}$ consequence of the *conditional independence* of Equation (5.8) in Section 5.4.3, the implied forecast probability for the hierarchical series $\hat{\mathbb{P}}_{[a,b]}$ is coherent and satisfies Definition 5.2. The proof first shows that $\hat{\mathbb{P}}_{[a]}$ is well defined, and then shows that HINT's aggregate marginal probability assigns a zero probability to any set that does not contain any coherent forecasts, which implies probabilistic coherence.

Lemma B.1: Aggregate Variables Coherence

Let $(\Omega_{[b]}, \mathcal{F}_{[b]}, \hat{\mathbb{P}}_{[b]})$ be a probabilistic forecast space, with $\mathcal{F}_{[b]}$ a σ-algebra on $\Omega_{[b]}$. The aggregation rule defines a probability measure over $\Omega_{[a]} = \mathbf{A}_{[a][b]}(\Omega_{[b]})$:

$$\hat{\mathbb{P}}_{[a]}(\mathbf{y}_{[a]}) = \int_{\Omega_{[b]}} \hat{\mathbb{P}}_{[b]}(\mathbf{y}_{[b]}) \mathbb{1}\{\mathbf{y}_{[a]} = \mathbf{A}_{[a][b]}\mathbf{y}_{[b]}\} d\mathbf{y}_{[b]} \tag{B.1}$$

Proof. We prove $\hat{\mathbb{P}}_{[a]}$ satisfies Kolmogorov axioms on $(\Omega_{[a]}, \mathcal{F}_{[a]}, \hat{\mathbb{P}}_{[a]})$ with $\Omega_{[a]} = \mathbf{A}_{[a][b]}(\Omega_{[b]})$.

1. $\hat{\mathbb{P}}_{[a]}(\mathcal{A}) \geq 0 \quad \forall \mathcal{A} \in \mathcal{F}_{[a]}$: This follows from the positivity of $\hat{\mathbb{P}}_{[b]}(\mathcal{B})$ and the indicator function.
2. $\hat{\mathbb{P}}_{[a]}(\Omega_{[a]}) = 1$: The unit measure assumption holds because

$$\hat{\mathbb{P}}_{[a]}(\mathbf{A}_{[a][b]}(\Omega_{[b]}) = \int_{\Omega_{[a]}} \int_{\Omega_{[b]}} \hat{\mathbb{P}}_{[b]}(\mathbf{y}_{[b]}) \mathbb{1}\{\mathbf{y}_{[a]} = \mathbf{A}_{[a][b]}\mathbf{y}_{[b]}\} d\mathbf{y}_{[b]} d\mathbf{y}_{[a]} = \int_{\Omega_{[b]}} \hat{\mathbb{P}}_{[b]}(\mathbf{y}_{[b]}) d\mathbf{y}_{[b]} = 1$$

3. $\hat{\mathbb{P}}_{[a]}\left(\bigcup_{i=1}^{\infty} \mathcal{A}_i\right) = \sum_{i=1}^{\infty} \hat{\mathbb{P}}(\mathcal{A}_i)$ for disjoint sets $\mathcal{A}_i$'s: The σ-additivity assumption holds

$$\begin{aligned}\hat{\mathbb{P}}_{[a]}\left(\bigcup_{i=1}^{\infty} \mathcal{A}_i\right) &= \hat{\mathbb{P}}_{[a]}\left(\bigcup_{i=1}^{\infty} \mathbf{A}_{[a][b]}(\mathcal{B}_i)\right) = \int \hat{\mathbb{P}}_{[b]}\left(\bigcup_{i=1}^{\infty} \mathcal{B}_i\right) \mathbb{1}\{\mathbf{y}_{[a]} = \mathbf{A}_{[a][b]}\mathbf{y}_{[b]}\} d\mathbf{y}_{[b]} \\ &= \int \hat{\mathbb{P}}_{[b]}\left(\bigcup_{i=1}^{\infty} \mathcal{B}_i\right) d\mathbf{y}_{[b]} = \sum_{i=1}^{\infty} \hat{\mathbb{P}}(\mathcal{B}_i) = \sum_{i=1}^{\infty} \hat{\mathbb{P}}(\mathcal{A}_i)\end{aligned}$$

□

Lemma B.2: Characterization of Probabilistic Coherence

Let $(\Omega_{[b]}, \mathcal{F}_{[b]}, \hat{\mathbb{P}}_{[b]})$ be a probabilistic forecast space, with $\mathcal{F}_{[b]}$ a σ-algebra on $\Omega_{[b]}$. If a forecast distribution assigns a zero probability to sets that don't contain coherent forecasts, it defines a coherent probabilistic forecast space $(\Omega_{[a,b]}, \mathcal{F}_{[a,b]}, \hat{\mathbb{P}}_{[a,b]})$ with $\Omega_{[a,b]} = \mathbf{S}_{[a,b][b]}(\Omega_{[b]})$.

$$\hat{\mathbb{P}}_{[a]}\left(\mathbf{y}_{[a]} \notin \mathbf{A}_{[a][b]}(\mathcal{B}) \mid \mathcal{B}\right) = 0 \implies \hat{\mathbb{P}}_{[a,b]}\left(\mathbf{S}_{[a,b][b]}(\mathcal{B})\right) = \hat{\mathbb{P}}_{[b]}\left(\mathcal{B}\right) \quad \forall \mathcal{B} \in \mathcal{F}_{[b]} \tag{B.2}$$

Proof.

$$\begin{aligned}\hat{\mathbb{P}}_{[a,b]}\left(\mathbf{S}_{[a,b][b]}(\mathcal{B})\right) &= \hat{\mathbb{P}}_{[a,b]}\left(\begin{bmatrix}\mathbf{A}_{[a][b]}\\ \mathbf{I}_{[b][b]}\end{bmatrix}(\mathcal{B})\right) = \hat{\mathbb{P}}_{[a,b]}\left(\left\{\begin{bmatrix}\mathbf{A}_{[a][b]}(\mathcal{B})\\ \mathbb{R}^{N_b}\end{bmatrix}\right\} \cap \left\{\begin{bmatrix}\mathbb{R}^{N_a}\\ \mathcal{B}\end{bmatrix}\right\}\right)\\ &= \hat{\mathbb{P}}_{[a]}\left(\mathbf{A}_{[a][b]}(\mathcal{B}) \mid \mathcal{B}\right)\hat{\mathbb{P}}_{[b]}\left(\mathcal{B}\right) = \left(1 - \hat{\mathbb{P}}_{[a]}\left(\mathbf{y}_{[a]} \notin \mathbf{A}_{[a][b]}(\mathcal{B}) \mid \mathcal{B}\right)\right) \times \hat{\mathbb{P}}_{[b]}\left(\mathcal{B}\right) = \hat{\mathbb{P}}_{[b]}\left(\mathcal{B}\right)\end{aligned}$$

The first equality is the image of a set $\mathcal{B} \in \Omega_{[b]}$ corresponding the constraints matrix transformation, the second equality defines the spanned space as a subspace intersection of the aggregate series and the bottom series, the third equality uses conditional probability multiplication rule, the final equality uses the zero probability assumption. □

Theorem B.1: HINT's probabilistic coherence.

With $(\Omega_{[b]}, \mathcal{F}_{[b]}, \hat{\mathbb{P}}_{[b]})$ probabilistic forecast space, we can construct a coherent probabilistic forecast space $(\Omega_{[a,b]}, \mathcal{F}_{[a,b]}, \hat{\mathbb{P}}_{[a,b]})$ with Lemma B.1's aggregation.

Proof. It follows from Section 5.4.3's *aggregation rule* that $\hat{\mathbb{P}}_{[a]}\left(\mathbf{y}_{[a]} \notin \mathbf{A}_{[a][b]}(\mathcal{B}) \mid \mathcal{B}\right) = 0$, using Lemma B.2 we obtain a probabilistic coherent space $(\Omega_{[a,b]}, \mathcal{F}_{[a,b]}, \hat{\mathbb{P}}_{[a,b]})$. □

B.2 Covariance Formula

Here we present the derivation of the covariance formula in Equation (5.11).

Proof. Using the law of total covariance, we get

$$\mathrm{Cov}(Y_{\beta,\tau}, Y_{\beta',\tau'}) = \mathbb{E}\left[\mathrm{Cov}(Y_{\beta,\tau}, Y_{\beta',\tau'} | \lambda_{\beta,\kappa,\tau}, \lambda_{\beta',\kappa,\tau'})\right] + \mathrm{Cov}\left(\mathrm{E}\left[Y_{\beta,\tau} | \lambda_{\beta,\kappa,\tau}\right], \mathbb{E}\left[Y_{\beta',\tau'} | \lambda_{\beta',\kappa,\tau'}\right]\right) \tag{B.3}$$

Using the conditional independence from Equation (5.8)). We can rewrite the expectation of the conditional covariance:

$$\begin{aligned} \mathbb{E}\left[\mathrm{Cov}(Y_{\beta,\tau}, Y_{\beta',\tau'} | \lambda_{\beta,\kappa,\tau}, \lambda_{\beta',\kappa,\tau'})\right] &= \mathbb{E}\left[\mathrm{Var}(Y_{\beta,\tau} | \lambda_{\beta,\kappa,\tau})\right] \mathbb{1}(\beta = \beta') \mathbb{1}(\tau = \tau') \\ &= \mathbb{E}\left[\lambda_{\beta,\kappa,\tau}\right] \mathbb{1}(\beta = \beta') \mathbb{1}(\tau = \tau') \\ &= \overline{\lambda}_{\beta,\tau} \mathbb{1}(\beta = \beta') \mathbb{1}(\tau = \tau') \end{aligned} \tag{B.4}$$

where $\overline{\lambda}_{\beta,\tau} = \sum_{\kappa=1}^{N_k} w_\kappa \lambda_{\beta,\kappa,\tau}$.

In the second term, because the conditional distributions are Poisson we have

$$\mathrm{E}\left[Y_{\beta,\tau} | \lambda_{\beta,\kappa,\tau}\right] = \lambda_{\beta,\kappa,\tau} \quad \text{and} \quad \mathrm{E}\left[Y_{\beta',\tau'} | \lambda_{\beta',\kappa,\tau'}\right] = \lambda_{\beta',\kappa,\tau'}$$

Which implies

$$\mathrm{Cov}\left(\mathbb{E}\left[Y_{\beta,\tau} | \lambda_{\beta,\kappa,\tau}\right], \mathbb{E}\left[Y_{\beta',\tau'} | \lambda_{\beta',\kappa,\tau'}\right]\right) = \sum_{\kappa=1}^{N_k} w_\kappa \left(\lambda_{\beta,\kappa,\tau} - \overline{\lambda}_{\beta,\tau}\right)\left(\lambda_{\beta',\kappa,\tau'} - \overline{\lambda}_{\beta',\tau}\right) \tag{B.5}$$

Therefore, the covariance formula is:

$$\mathrm{Cov}(Y_{\beta,\tau}, Y_{\beta',\tau'}) = \overline{\lambda}_{\beta,\tau} \mathbb{1}(\beta = \beta') \mathbb{1}(\tau = \tau') + \sum_{\kappa=1}^{N_k} w_\kappa \left(\lambda_{\beta,\kappa,\tau} - \overline{\lambda}_{\beta,\tau}\right)\left(\lambda_{\beta',\kappa,\tau'} - \overline{\lambda}_{\beta',\tau'}\right)$$

□

B.3 Mixture Components and Covariance Matrix Rank

As mentioned in Section 5.4.5, complex correlations across series benefit from a higher number of mixture components. We ground this intuition on the expressiveness of HINT's Poisson Mixture covariance matrix controlled by its rank. We can show that for HINT's finite Poisson mixture distribution, the bottom level's estimator of the *non-diagonal* covariance matrix series is a matrix of rank at most $K-1$ given by:

$$\mathrm{Cov}(\mathbf{y}_{[b],\tau}) = \sum_{\kappa=1}^{K} \mathbf{w}_{\kappa}(\boldsymbol{\lambda}_{[b],\kappa,\tau} - \bar{\boldsymbol{\lambda}}_{[b],\tau})(\boldsymbol{\lambda}_{[b],\kappa,\tau} - \bar{\boldsymbol{\lambda}}_{[b],\tau})^{\intercal} \in \mathbb{R}^{N_b \times N_b} \tag{B.6}$$

Proof. One can easily extend the pair-wise covariance from Equation (B.5) to multivariate covariance Equation (B.6).

Let $\mathbf{z}_{\kappa} = (\boldsymbol{\lambda}_{[b],\kappa,\tau} - \bar{\boldsymbol{\lambda}}_{[b],\tau})$, by construction we can show that $\sum_{\kappa=1}^{K} \mathbf{z}_{\kappa} = 0$.
Rewriting the last vector $\mathbf{z}_K = -\sum_{\kappa=1}^{K-1} \mathbf{z}_k$ we obtain a sum of $K-1$ rank-1 matrices

$$\sum_{k=1}^{K} \mathbf{z}_k \mathbf{z}_k^{\intercal} = \sum_{k=1}^{K-1} \mathbf{z}_k \mathbf{z}_k^{\intercal} + \left(-\sum_{k=1}^{K-1} \mathbf{z}_k\right) \mathbf{z}_K^{\intercal} = \sum_{k=1}^{K-1} \mathbf{z}_k (\mathbf{z}_k - \mathbf{z}_K)^{\intercal}$$

which implies that HINT's modeled covariance matrix rank is upper bounded by $K-1$. □

B.4 Dataset Details

The `Traffic` dataset, as mentioned, measures the occupancy rates of 963 freeway lanes from the Bay Area. The original data is at a 10-minute frequency from January 1st 2008 to March 30th 2009. The dataset is further aggregated from the 10-minute frequency into daily frequency with 366 observations. We match the sample procedure from previous hierarchical forecasting literature (Souhaib and Bonsoo, 2019; Rangapuram et al., 2021), and use the same 200 bottom level series from the 963 available. From these 200 bottom level series a hierarchy is randomly defined by aggregating them into quarters and halves of 50 and 100 series each, finally we consider the total aggregation. Table B.1 describes the hierarchical structure.

Table B.1: San Francisco Bay Area highway traffic data summary.
[*] The hierarchical structure is randomly defined.

Geographical Level [*]	Series per Level
Bay Area	1
Halves	2
Quarters	4
Bottom	200
Total	207

The `Tourism-L` dataset contains 555 monthly series from 1998 to 2016, it is organized by geography and purpose of travel. The four-level geographical hierarchy comprises seven states, divided further into 27 zones and 76 regions. The categories for purpose of travel are holiday, visiting friends and relatives, business and other. This dataset has been referenced by important hierarchical forecasting studies like the one of the `MinT` reconciliation strategy and the more recent `HierE2E` (Wickramasuriya et al., 2019; Rangapuram et al., 2021). `Tourism-L` is a grouped dataset, it has two dimensions in which it is aggregated, the total level aggregation and its four associated purposes. Table B.2 describes the group and hierarchical structures.

Table B.2: Australian Tourism flows.

Geographical Level	Series per Level	Series per Level & Purpose	Total
Australia	1	4	5
States	7	28	35
Zones	27	108	135
Regions	76	304	380
Total	111	444	555

The `Favorita` dataset, once balanced for items and stores, contains 217,944 bottom level series, in contrast the original competition considers 210,645 series. We resort for this balance because, for the moment the `GroupBU` version of the Poisson Mixture requires balanced hierarchies for its estimation. In the case of the `Favorita` experiment we consider a geographical hierarchy (93 nodes) conditional of each of grocery item (4,036). The hierarchy defines 153,368 new aggregate series at the item-country, item-state, and item-city levels. Table B.3 describes the structure.

Regarding the dataset preprocessing, we confirmed observations from the best submissions to the Kaggle competition. Most holiday distances included in the dataset and covariates like oil production lack value for the forecasts. The models did not benefit from a long history, filtering the training window to the 2017 year consistently produced better results.

Table B.3: Favorita Grocery Sales.

Geographical Level	Nodes per Level	Series per Level	Total
Ecuador	1	4,036	4,036
States	16	64,576	64,576
Cities	22	88,792	88,792
Stores	54	217,944	217,944
Total	93	371,312	371,312

B.5 Poisson Mixture Size Ablation Study

As shown in Appendix B.2, the job of HINT' mixture distribution goes beyond forecasting single time series but also modeling correlations across them too. Based on the covariance matrix expressiveness theoretical properties, it is reasonable to expect that the number of optimal components grows with the complexity of the modeled time series structure.

In this ablation study, we empirically test these intuitions showing how the number of optimally selected HINT's mixture components grows with larger datasets. For the experiment, we measured the cross-validation performance of HINT' configurations as defined in Table 5.2 explored automatically with HYPEROPT (Bergstra et al., 2011) where we fix the number of mixture components.

Table B.4: Empirical evaluation of probabilistic coherent forecasts for different HINT-`GroupBU`, varying the Poisson mixture size. Mean overall *scaled continuous ranked probability score* (sCRPS). The best result is highlighted (lower measurements are preferred).

Dataset	Level	$K = 1$	$K = 10$	$K = 25$	$K = 50$	$K = 100$
`Traffic`	Overall	0.1647 ± 0.0009	0.1435 ± 0.0947	$\mathbf{0.0958 \pm 0.0005}$	0.1337 ± 0.0004	0.1261 ± 0.0037
`Tourism-L`	Overall	0.1673 ± 0.0052	0.1380 ± 0.0017	$\mathbf{0.1247 \pm 0.0025}$	0.1284 ± 0.0031	0.1251 ± 0.0034
`Favorita`	Overall	0.8390 ± 0.0124	0.4416 ± 0.0152	$--$	0.4204 ± 0.0108	$\mathbf{0.3758 \pm 0.0040}$

Table B.4 reports the validation probabilistic forecast accuracy measured with sCRPS, across `Traffic`, `Tourism-L`, and `Favorita`, for HINT-GroupBU with different number of Poisson mixture components. For this experiment, we report the overall validation sCRPS averaged over four independent HYPEROPT runs with twelve optimization steps and eight steps in the `Traffic` dataset.

Table B.4's sCRPS measurements suggest that there is a *bias-variance trade-off* controlled by the Poisson mixture size. When $K = 1$, HINT-GroupBU model corresponds to Poisson regression and treats each series as probabilistically independent, such model high-bias simple model produced forecasts with the worst sCRPS. The forecast accuracy improves as the number of Poisson components increases from $K = 1$, but the accuracy begins to deteriorate beyond a certain threshold. We hypothesize that a small number of mixture components does not have enough degrees of freedom to describe the data, and too many mixture components lead to over-fitting the training data, resulting in large variance on the validation data.

We observed that the precise value of an optimal Poisson mixture components varies across the datasets. Larger datasets, or datasets with a complex time series correlation structure, appear to benefit from more flexible probability mixtures. `Traffic`, our smallest dataset, produced optimal results with $K = 25$ components, `Tourism-L`, a medium-sized dataset, produced optimal results with $K = 25$ components. Finally `Favorita` our largest dataset, did not saturate even with the largest number of components we experimented with; we capped the choice of the number of mixture components at $K = 100$ due to GPU memory constraints.

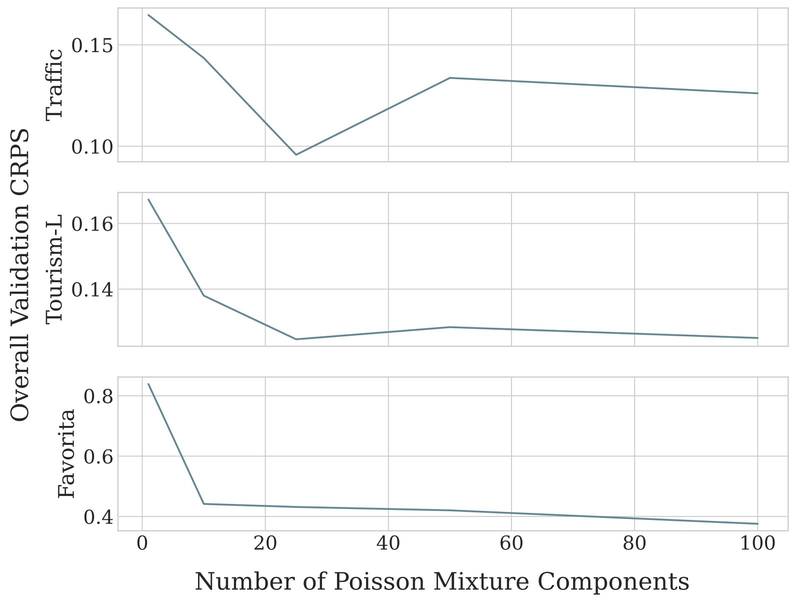


Figure B.1: Poisson Mixture size ablation study. There is a bias-variance trade-off controlled by the mixture components, both `Traffic` and `Tourism-L` have an optimal values of 25 components, beyond which the sCRPS validation performance worsens, with classic U-shaped pattern. In the case of `Favorita`, the largest dataset, the validation sCRPS continued to improve beyond $K = 100$.

B.6 Model Parameter Details

Table B.5: *Hierarchical Forecast Network* (HINT) architecture parameters configured once per dataset. These hyperparameters correspond to the first selection phase preceding the automatic optimization.
[*] SGD batch follows an ablation study considering values between {2,4,8,16,32,64,100}. We report the best validation batch size.

Parameter	Notation	Considered Values		
		`Traffic`	`Tourism-L`	`Favorita`
SGD Batch Size[*] .	-	4	4	100
Activation Function.	-	PeLU	PeLU	PeLU
Temporal Convolution Kernel Size.	N_{ck}	$\{7\}$	$\{2\}$	$\{2\}$
Temporal Convolution Layers.	N_{cl}	$\{3\}$	$\{5\}$	$\{5\}$
Temporal Convolution Filters.	N_p	$\{10\}$	$\{30\}$	$\{30\}$
Future Encoder Dimension.	N_f	$\{50\}$	$\{50\}$	$\{50\}$
Static Encoder Dimension.	N_s	$\{100\}$	$\{100\}$	$\{100\}$
Horizon Agnostic Decoder Dimensions.	N_{ag}	$\{50\}$	$\{50\}$	$\{50\}$
Horizon Specific Decoder Dimensions.	N_{sp}	$\{20\}$	$\{20\}$	$\{20\}$
Poisson Mixture Weight Decoder Layers.	N_{wdl}	$\{3\}$	$\{4\}$	$\{4\}$
Poisson Mixture Rate Decoder Layers.	N_{rdl}	$\{2\}$	$\{3\}$	$\{3\}$
Poisson Mixture Components.	N_k	$\{25\}$	$\{25\}$	$\{100\}$
GPU Training Configuration.	-	2 x NVIDIA V100	2 x NVIDIA V100	4 x NVIDIA V100

As mentioned in Section 5.5.5, for the overall hyperparameter selection, we used a standard two-stage approach where we fixed the architecture, the probability distribution to estimate (and implicit training loss), and a second stage where we optimized the architecture's training procedure.

In the first stage, we carefully fixed the architecture and the probability distribution to estimate. The most important heuristic guiding this selection was to increase the architecture's and probability's capacity for larger or more complex datasets. To increase the network's capacity, we increased the number of convolution layers N_{cl} and convolution filters N_p as well as the mixture weight and rate decoder layers N_{wdl}, N_{rdl}. In particular, since `Traffic` is the smallest dataset, we opted for a reduced model size and the number of Poisson mixture components to control the model's variability. Additionally, due to the dataset's strong weekly seasonality pattern, we adjusted the convolution kernel size to encompass seven days. We control the probability's capacity with the mixture size and SGD batchsize. In an ablation study similar to Appendix B.5 we found that for datasets with strong correlations like `Favorita` and `Tourism-L` maximizing the batch size with respect to GPU memory limitations resulted in better validation performance; for `Traffic`, though the entire dataset could fit in memory at once, it was preferable to feed in subsets to allow the model to learn from different randomly sampled highway groups in each epoch.

For the second hyperparameter selection stage, as reported in Section 5.5.5, for each fixed architecture and probability we optimally explored its training procedure hyperparameters defined in Table 5.2 using HYPEROPT algorithm's Bayesian optimization (Bergstra et al., 2011). The second phase only considers the optimal exploration of the learning rate, random initialization, and the number of SGD epochs, the selection is guided by temporal cross-validation signal obtained from the dataset partition introduced in Section 5.5.3.

B.7 Qualitative Analysis of Poisson Mixture Rates

We use a histogram in Figure B.2 to visualize the distribution of Poisson Mixture rates learned by the HINT-GroupBU method on the Traffic, Tourism-L, and Favorita datasets. These rates correspond to those of the models from Table 5.3, for a base time series and a single time step ahead in the forecast horizon. In the histogram, we use the learned mixture weights w_k in the vertical axis to account for the probability of each Poisson rate λ_k represented in the horizontal axis.

Exciting patterns arise from visualizing the learned Poisson Mixture rates and their weights. First, the HINT likelihood, in similar fashion to Zero Inflated Poisson Regression (ZIP) (Lambert, 1992), is capable of modeling sparse data in the case of the Favorita bottom level series; as we can see from the probability accumulation around zero, this may explain its superior performance on Favorita bottom level series that tends to be sparse as previously shown in Figure 5.6. Second, all the Poisson rate distributions show multiple modes, a quality that cannot be replicated by a Gaussian, uni-modal, and symmetric distribution. This observation further motivates using a flexible distribution capable of better modeling the underlying data generation processes. Third, the distribution of Poisson rates in the Tourism-L dataset can overcome the limitations of simple Poisson distributions that tend to have collapsed variances for aggregated series due to their scale since the variance of the series can still be correctly modeled by the variance of the Poisson rates.

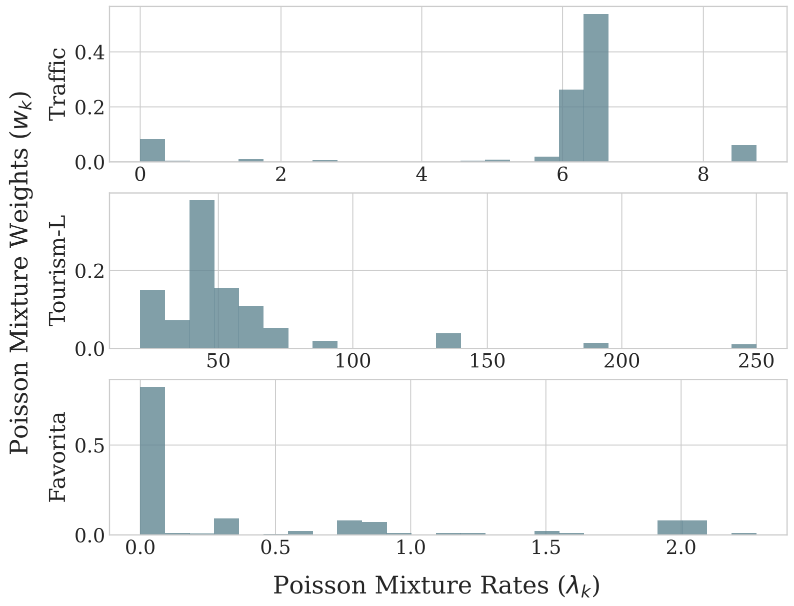


Figure B.2: Poisson Mixture rates distribution for the HINT-GroupBU model reported in Table 5.3. A single bottom-level series and a time step ahead in the forecast horizon is considered. The mixture distribution is capable of flexibly modeling multimodal processes, overcome Poisson regression's limitations for aggregated data and model zero inflated processes for disaggregated data. These qualities make it exceptionally useful for hierarchical forecasting tasks.

Appendix C

Long Multi-horizon Forecasting

C.1 Neural Basis Approximation Theorem

In this Appendix we prove the *neural basis expansion approximation theorem* introduced in Section 6.5.3. We show that NHITS' hierarchical interpolation can arbitrarily approximate infinitely long horizons ($\tau \in [0,1]$ continuous horizon), as long as the interpolating functions g are defined by a projections to informed multi-resolution functions, and the forecast relationships satisfy smoothness conditions. We prove the case when $g_{w,h}(\tau) = \theta_{w,h}\phi_{w,h}(\tau) = \theta_{w,h}\mathbb{1}\{\tau \in [2^{-w}(h-1), 2^{-w}h]\}$ are piecewise constants and the inputs $\mathbf{y}_{[t-L:t]} \in [0,1]$. The proof for linear, spline functions and $\mathbf{y}_{[t-L:t]} \in [a,b]$ is analogous.

Lemma C.1: Haar's Wavelets Approximation

Let a function representing an infinite forecast horizon be $\mathcal{Y} : [0,1] \rightarrow \mathbb{R}$ a square integrable function $\mathcal{L}^2([0,1])$. The forecast function $\mathcal{Y}$ can be arbitrarily well approximated by a linear combination of piecewise constants:

$$V_w = \{\phi_{w,h}(\tau) = \phi(2^w(\tau - h)) \mid w \in \mathbb{Z}, h \in 2^{-w} \times [0, \dots, 2^w]\}$$

where $w \in \mathbb{N}$ controls the frequency/indicator's length and h the time-location (knots) around which the indicator $\phi_{w,h}(\tau) = \mathbb{1}\{\tau \in [2^{-w}(h-1), 2^{-w}h]\}$ is active. That is, $\forall \epsilon > 0$, there is a $w \in \mathbb{N}$ and $\hat{\mathcal{Y}}(\tau|\mathbf{y}_{[t-L:t]}) = \mathrm{Proj}_{V_w}(\mathcal{Y}(\tau|\mathbf{y}_{[t-L:t]})) \in \mathrm{Span}(\phi_{w,h})$ such that

$$\int_{[0,1]} |\mathcal{Y}(\tau) - \hat{\mathcal{Y}}(\tau)| d\tau = \int_{[0,1]} |\mathcal{Y}(\tau) - \sum_{w,h} \theta_{w,h}\phi_{w,h}(\tau)| d\tau \leq \epsilon \tag{C.1}$$

Proof. This classical proof can be traced back to Haar's work (1910). The indicator functions $V_w = \{\phi_{w,h}(\tau)\}$ are also referred in literature as Haar scaling functions or father wavelets. Details provided in Boggess and Narcowich, 2015.Let the number of coefficients for the ϵ-approximation $\hat{\mathcal{Y}}(\tau|\mathbf{y}_{[t-L:t]})$ be denoted as $N_\epsilon = \sum_{i=0}^{w} 2^i$. □

Lemma C.2: Neural Network's Universal Approximation

Let a forecast mapping $\mathcal{Y}(\cdot \mid \mathbf{y}_{[t-L:t]}) : [0,1]^L \rightarrow \mathcal{L}^2([0,1])$ be ϵ-approximated by $\hat{\mathcal{Y}}(\tau|\mathbf{y}_{[t-L:t]}) = \mathrm{Proj}_{V_w}(\mathcal{Y}(\tau|\mathbf{y}_{[t-L:t]}))$, the projection to multi-resolution piecewise constants. If the relationship between $\mathbf{y}_{[t-L:t]} \in [0,1]^L$ and $\theta_{w,h}$ varies smoothly, for instance $\theta_{w,h} : [0,1]^L \rightarrow \mathbb{R}$ is a K-Lipschitz function then for all $\epsilon > 0$ there exists a three-layer neural network $\hat{\theta}_{w,h} : [0,1]^L \rightarrow \mathbb{R}$ with $O\left(L(\frac{K}{\varepsilon})^L\right)$ neurons and ReLU activations such that

$$\int_{[0,1]^L} |\theta_{w,h}(\mathbf{y}_{[t-L:t]}) - \hat{\theta}_{w,h}(\mathbf{y}_{[t-L:t]})| d\mathbf{y}_{[t-L:t]} \leq \epsilon \tag{C.2}$$

Proof. This lemma is a special case of the neural universal approximation theorem that states the approximation capacity of neural networks of arbitrary width (Hornik, 1991). The theorem has refined versions where the width can be decreased under more restrictive conditions for the approximated function (Barron, 1993; Hanin and Sellke, 2017). □

Theorem C.1: Neural Basis Approximation

Theorem 1. Let a forecast mapping be

$\mathcal{Y}(\cdot \mid \mathbf{y}_{[t-L:t]}) : [0,1]^L \rightarrow \mathcal{F}$, where the forecast functions $\mathcal{F} = \{\mathcal{Y}(\tau) : [0,1] \rightarrow \mathbb{R}\} = \mathcal{L}^2([0,1])$ representing a continuous horizon, are square integrable.

If the multi-resolution functions V_w can arbitrarily approximate $\mathcal{L}^2([0,1])$. And the projection $\text{Proj}_{V_w}(\mathcal{Y}(\tau))$ varies smoothly on $\mathbf{y}_{[t-L:t]}$. Then the forecast mapping $\mathcal{Y}(\cdot \mid \mathbf{y}_{[t-L:t]})$ can be arbitrarily approximated by a neural network learning a finite number of multi-resolution coefficients $\hat{\theta}_{w,h}$.

That is $\forall \epsilon > 0$,

$$\begin{aligned} &\int |\mathcal{Y}(\tau \mid \mathbf{y}_{[t-L:t]}) - \tilde{\mathcal{Y}}(\tau \mid \mathbf{y}_{[t-L:t]})| d\tau \\ &\qquad = \int |\mathcal{Y}(\tau \mid \mathbf{y}_{[t-L:t]}) - \sum_{w,h} \hat{\theta}_{w,h}(\mathbf{y}_{[t-L:t]}) \phi_{w,h}(\tau)| d\tau \leq \epsilon \end{aligned} \tag{C.3}$$

Proof. For simplicity of the proof, we will omit the conditional lags $\mathbf{y}_{[t-L:t]}$. Using both the neural approximation $\tilde{\mathcal{Y}}$ from Lemma 2, and Haar's approximation $\hat{\mathcal{Y}}$ from Lemma 1,

$$\int |\mathcal{Y}(\tau) - \tilde{\mathcal{Y}}(\tau)| d\tau = \int |(\mathcal{Y}(\tau) - \hat{\mathcal{Y}}(\tau)) + (\hat{\mathcal{Y}}(\tau) - \tilde{\mathcal{Y}}(\tau))| d\tau$$

By the triangular inequality:

$$\begin{aligned} \int |\mathcal{Y}(\tau) - \tilde{\mathcal{Y}}(\tau)| d\tau \leq &\int |\mathcal{Y}(\tau) - \hat{\mathcal{Y}}(\tau)| \\ &+ |\sum_{w,h} \theta_{w,h}\phi_{w,h}(\tau) - \sum_{w,h} \hat{\theta}_{w,h}\phi_{w,h}(\tau)| d\tau \end{aligned}$$

By a special case of Fubini's theorem

$$\begin{aligned} &\int |\mathcal{Y}(\tau) - \tilde{\mathcal{Y}}(\tau)| d\tau \leq \\ &\int |\mathcal{Y}(\tau) - \sum_{w,h} \hat{\mathcal{Y}}(\tau)| d\tau + \sum_{w,h} \int_{\tau} |(\theta_{w,h} - \hat{\theta}_{w,h})\phi_{w,h}(\tau)| d\tau \end{aligned}$$

Using positivity and bounds of the indicator functions

$$\int |\mathcal{Y}(\tau) - \tilde{\mathcal{Y}}(\tau)|d\tau \leq$$
$$\int_\tau |\mathcal{Y}(\tau) - \sum_{w,h} \hat{\mathcal{Y}}(\tau)|d\tau + \sum_{w,h} |\theta_{w,h} - \hat{\theta}_{w,h}| \int_\tau \phi_{w,h}(\tau)d\tau$$
$$< \int_\tau |\mathcal{Y}(\tau) - \sum_{w,h} \hat{\mathcal{Y}}(\tau)|d\tau + \sum_{w,h} |\theta_{w,h} - \hat{\theta}_{w,h}|$$

To conclude we use the both arbitrary approximations from the Haar projection and the approximation to the finite multi-resolution coefficients

$$\int |\mathcal{Y}(\tau) - \tilde{\mathcal{Y}}(\tau)|d\tau \leq$$
$$\int |\mathcal{Y}(\tau) - \hat{\mathcal{Y}}(\tau)|d\tau + \sum_{w,h} |\theta_{w,h} - \hat{\theta}_{w,h}| \leq \epsilon_1 + N_{\epsilon_1}\epsilon_2 \ \leq \epsilon$$

□

C.2 Computational Complexity Analysis

We consider a single forecast of length H for the following complexity analysis, with a NBEATS and a NHITS architecture of B blocks. We do not consider the batch dimension. We consider most practical situations, the input size $L = \mathcal{O}(H)$ linked to the horizon length.

The block operation described by Equation (6.2) has complexity dominated by the fully connected layers of $\mathcal{O}(H\,N_h)$, with N_h the number of hidden units that we treat as a constant. The depth of stacked blocks in the NBEATS-G architecture, that endows it with its expressivity, is associated to a computational complexity that scales linearly $\mathcal{O}(HB)$, with B the number of blocks.

In contrast the NHITS architecture that specializes each stack in different frequencies, through the expressivity ratios, can greatly reduce the amount of parameters needed for each layer. When we use *exponentially increasing expressivity* ratios through the depth of the architecture blocks it allows to model complex dependencies, while controlling the number of parameters used on each output layer. If the *expressivity ratio* is defined as $r_\ell = r^l$ then the space complexity of NHITS scales geometrically $\mathcal{O}(\sum_{l=0}^{B} Hr^l) = \mathcal{O}\left((H(1-r^B)/(1-r)\right)$.

Table C.1: Computational complexity of neural based forecasting methods as a function of the output size H. For simplicity, we assume that the input size L scales linearly with respect to H. For NHITS and NBEATS we also consider the network's B blocks.

Model	Time	Memory
LSTM	$\mathcal{O}(H)$	$O(H)$
ESRNN	$\mathcal{O}(H)$	$O(H)$
TCN	$\mathcal{O}(H)$	$O(H)$
Transformer	$\mathcal{O}(H^2)$	$O(H^2)$
Reformer	$\mathcal{O}(H\,logH)$	$O(H\,logH)$
Informer	$\mathcal{O}(H\,logH)$	$O(H\,logH)$
Autoformer	$\mathcal{O}(H\,logH)$	$O(H\,logH)$
LogTrans	$\mathcal{O}(H\,logH)$	$O(H^2)$
NBEATS-I	$\mathcal{O}(H^2B)$	$O(H^2B)$
NBEATS-G	$\mathcal{O}(HB)$	$O(HB)$
NHITS	$\mathcal{O}(H(1-r^B)/(1-r))$	$O(H(1-r^B)/(1-r))$

C.3 Hyperparameter Exploration

All benchmark neural forecasting methods optimize the length of the input $\{96, 192, 336, 720\}$ for ETT, Weather, and ECL, $\{24, 36, 48, 60\}$ for ILI, and $\{24, 48, 96, 192, 288, 480, 672\}$ for ETTm. The Transformer-based models: Autoformer, Informer, LogTrans, and Reformer are trained with MSE loss and ADAM of 32 batch size, using a starting learning rate of 1e-4, halved every two epochs, for ten epochs with early stopping. Additionally, for comparability of the computational requirements, all use two encoder layers and one decoder layer.

We use the adaptation to the long-horizon time series setting provided by Wu et al. 2021 of the Reformer (Kitaev et al., 2020), and LogTrans (Li et al., 2019), with the multi-step forecasting strategy (non-dynamic decoding).

The Autoformer (Wu et al., 2021) explores with grid-search the top-k auto-correlation filter hyper-parameter in $\{1, 2, 3, 4, 5\}$. And fixes inputs $L = 96$ for all datasets except for ILI in which they use $L = 36$. For the Informer (Zhou et al., 2020) we use the reported best hyperparameters found using an grid-search, that include dimensions of the encoder layers $\{6, 4, 3, 2\}$, the dimension of the decoder layer $\{2\}$, the heads of the multi-head attention layers $\{8, 16\}$ and its output's $\{512\}$.

We considered other classic models, like the automatically selected ARIMA model (Hyndman and Khandakar, 2008). The method is trained with maximum likelihood estimation under normality and independence. And integrates root statistical tests with model selection performed with Akaike's Information Criterion.

Finally, as mentioned in Section 6.5.3 for NHITS main results we limit the exploration to a minimal space of hyperparameters. We only consider the kernel pooling size for multi-rate sampling from Equation (6.1), the number of coefficients in Equation (6.2) and the random seed from Table 6.2.

C.4 Univariate Forecasting

As a complement of the main results from Section 6.5.4, in this Appendix, we performed univariate forecasting experiments for the ETTm$_2$ and Exchange datasets. This experiment allows us to compare closely with other methods specialized in long-horizon forecasting that also considered this setting (Zhou et al., 2020; Wu et al., 2021).

For the univariate setting, we consider the Transformer-based (1) Autoformer (Wu et al., 2021), (2) Informer (Zhou et al., 2020), (3) LogTrans (Li et al., 2019) and (4) Reformer (Kitaev et al., 2020) models. We selected other well-established univariate forecasting benchmarks: (5) NBEATS (Oreshkin et al., 2020), (6) DeepAR (Salinas et al., 2020) model, which takes autoregressive features and combines them with classic recurrent networks. (7) Prophet (Taylor and Letham, 2018), an additive regression model that accounts for different frequencies non-linear trends, seasonal and holiday effects and (8) an auto ARIMA (Hyndman and Khandakar, 2008).

Table C.2 summarizes the univariate forecasting results. NHITS significantly improves over the alternatives, decreasing 17% in MAE and 25% in MSE across datasets and horizons, with respect the best alternative. As noticed by the community recurrent based strategies like the one from ARIMA, tend to degrade due to the concatenation of errors phenomenon.

Table C.2: Empirical evaluation of long multi-horizon **univariate** forecasts. *Mean Absolute Error* (MAE) and *Mean Squared Error* (MSE) for predictions averaged over eight runs, the best result is highlighted in bold (lower is better). We gradually prolong the forecast horizon.

		NHITS		Autoformer		Informer		Reformer		NBEATS		DeepAR		ARIMA	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm$_2$	96	0.066	**0.185**	**0.065**	0.189	0.088	0.225	0.131	0.288	0.082	0.219	0.099	0.237	0.211	0.362
	192	**0.087**	**0.223**	0.118	0.256	0.132	0.283	0.186	0.354	0.120	0.268	0.154	0.310	0.261	0.406
	336	**0.106**	**0.251**	0.154	0.305	0.180	0.336	0.220	0.381	0.226	0.370	0.277	0.428	0.317	0.448
	720	**0.157**	**0.312**	0.182	0.335	0.300	0.435	0.267	0.430	0.188	0.338	0.332	0.468	0.366	0.487
Exchange	96	**0.093**	**0.223**	0.241	0.299	0.591	0.615	1.327	0.944	0.156	0.299	0.417	0.515	0.112	0.245
	192	**0.230**	**0.313**	0.273	0.665	1.183	0.912	1.258	0.924	0.669	0.665	0.813	0.735	0.304	0.404
	336	**0.370**	**0.486**	0.508	0.605	1.367	0.984	2.179	1.296	0.611	0.605	1.331	0.962	0.736	0.598
	720	**0.728**	**0.569**	0.991	0.860	1.872	1.072	1.280	0.953	1.111	0.860	1.890	1.181	1.871	0.935

C.5 Ablation Studies

This section performs ablation studies on the validation set of five datasets that share horizon lengths, ETTm$_2$, Exchange, ECL, Traffic-L, and Weather. The section's experiments control for NHITS settings described in Table 6.2, only varying a single characteristic of interest of the network and measuring the effects in validation.

C.5.1 Pooling Configurations

In Section 6.4.1 we described the *multi-rate signal sampling* enhancement of the NHITS architecture. Here we conduct a study to compare the accuracy effects of different pooling alternatives, on Equation (6.1). We consider the MaxPool and AveragePool configurations. As shown in Table C.3, the MaxPool operation consistently outperforms the AveragePool alternative, with MAE improvements up to 15% and MSE up to 8% in the most extended horizon. On average, the forecasting accuracy favors the MaxPool method across the datasets and horizons.

Table C.3: Empirical evaluation of long multi-horizon multivariate forecasts for NHITS with different pooling configurations. All other hyperparameters were kept constant across all datasets. MAE and MSE for predictions averaged over eight seeds, the best result is highlighted in bold (lower is better). Average percentage difference relative to average pooling in the last panel.

		MaxPool		AveragePool	
		MSE	MAE	MSE	MAE
ETTm$_2$	96	**0.185**	0.265	0.186	**0.262**
	192	**0.244**	**0.308**	0.257	0.315
	336	**0.301**	**0.347**	0.312	0.356
	720	**0.429**	**0.438**	0.436	0.447
ECL	96	**0.152**	**0.257**	0.181	0.290
	192	**0.172**	**0.275**	0.212	0.320
	336	**0.197**	**0.304**	0.238	0.343
	720	**0.248**	**0.347**	0.309	0.400
Exchange	96	**0.109**	**0.232**	0.112	0.238
	192	0.280	0.375	**0.265**	**0.371**
	336	**0.472**	0.504	0.501	**0.502**
	720	**1.241**	**0.823**	1.610	0.942
Traffic-L	96	**0.405**	**0.286**	0.468	0.332
	192	**0.421**	**0.297**	0.490	0.347
	336	**0.448**	**0.318**	0.531	0.371
	720	**0.527**	**0.362**	0.602	0.400
P. Diff.	96	**-8.911**	**-6.251**	0.000	0.000
	192	**-7.544**	**-6.085**	0.000	0.000
	336	**-8.740**	**-4.575**	0.000	0.000
	720	**-15.22**	**-8.318**	0.000	0.000

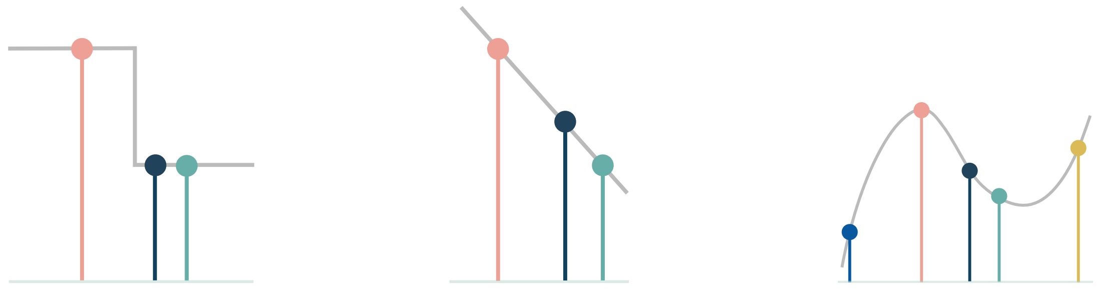


Figure C.1: Proposed interpolator configurations.

C.5.2 Interpolation Configurations

In Section 6.4.3 we described the *hierarchical interpolation* enhancement of the multi-step prediction strategy. Here we conduct a study to compare the accuracy effects of different interpolation alternatives. To do it, we change the interpolation technique used in the multi-step forecasting strategy of the NHITS architecture. The interpolation techniques considered are nearest neighbor, linear and cubic. We describe them in detail below.

Recalling the notation from Section 6.4.3, consider the time indexes of a multi-step prediction $\tau \in \{t+1,\dots,t+H\}$, let $\mathcal{T} = \{t+1, t+1+1/r_{\ell}\dots,t+H\}$ be the anchored indexes in NHITS layer ℓ, and the forecast $\hat{y}_{\tau,\ell} = g(\tau,\theta^{f}_{\ell})$ and backast $\tilde{y}_{\tau,\ell} = g(\tau,\theta^{f}_{\ell})$ components. Here we define different alternatives for the interpolating function $g \in \mathcal{C}^0, \mathcal{C}^1, \mathcal{C}^2$. For simplicity we skip the ℓ layer index.

Nearest Neighbor. In the simplest interpolation, we use the anchor observations in the time dimension closest to the observation we want to predict. Specifically, the prediction is defined as follows:

$$\hat{y}_{\tau} = \theta[t^{*}] \quad \text{with} \quad t^{*} = \text{argmin}_{t\in\mathcal{T}}\{|t-\tau|\} \tag{C.4}$$

Linear. An efficient alternative is the linear interpolation method, which uses the two closest neighbor indexes t_1 and t_1, and fits a linear function that passes through both.

$$\hat{y}_{\tau} = \left(\theta[t_1] + \left(\frac{\theta[t_2]-\theta[t_1]}{t_2-t_1}\right)(\tau - t_1)\right) \tag{C.5}$$

Cubic. Finally we consider the Hermite cubic polynomials defined by the interpolation constraints for two anchor observations θ_{t_1} and θ_{t_1} and its first derivatives $\theta^{'}_{t_1}$ and $\theta^{'}_{t_1}$.

$$\hat{y}_{\tau} = \theta[t_1]\phi_1(\tau) + \theta[t_2]\phi_2(\tau) + \theta^{'}[t_1]\psi_1(\tau) + \theta^{'}[t_2]\psi_2(\tau) \tag{C.6}$$

With the Hermite cubic basis defined by:

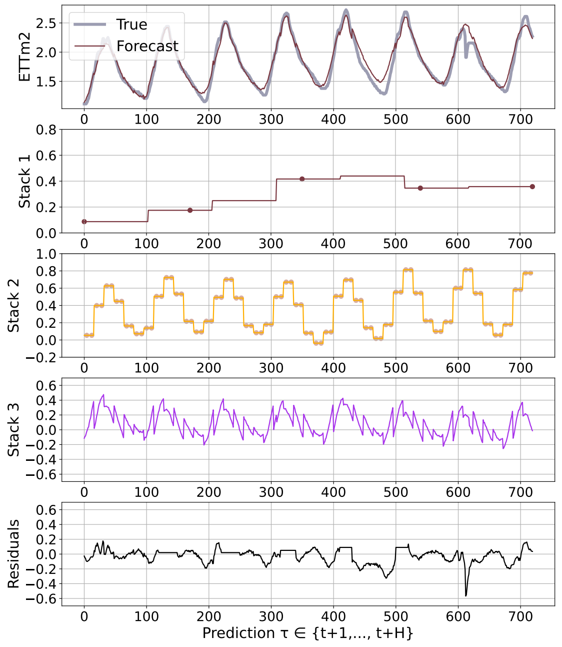
$$\phi_1(\tau) = 2\tau^3 - 3\tau^2 + 1 \tag{C.7a}$$
$$\phi_2(\tau) = -2\tau^3 + 3\tau^2 \tag{C.7b}$$
$$\psi_1(\tau) = \tau^3 - 2\tau^2 + \tau \tag{C.7c}$$
$$\psi_2(\tau) = \tau^3 - \tau^2 \tag{C.7d}$$

The ablation study results for the different interpolation techniques are summarized in Table C.4, we report the average MAE and MSE performance across the five datasets. Figure C.2 presents the decomposition for the different interpolation techniques. We found that linear and cubic interpolation consistently outperform the nearest neighbor alternative, and show monotonic improvements relative to the nearest neighbor technique along the forecasting horizon.

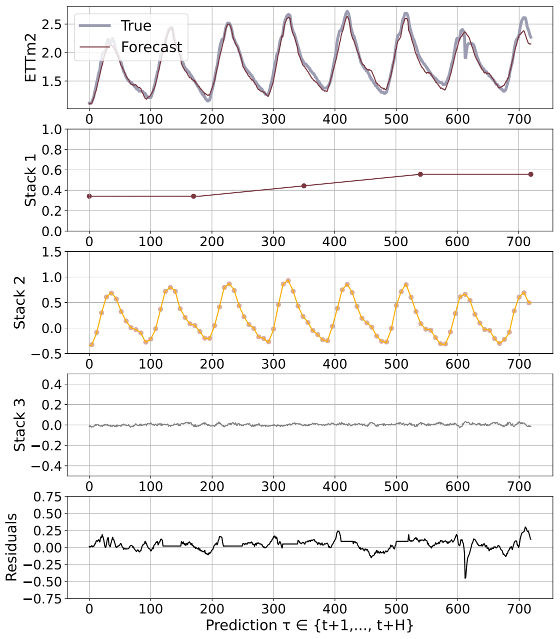
The linear interpolation improvements over nearest neighbors are up to 15.8%, and up to 7.0% for the cubic interpolation. When comparing between linear and cubic the results are inconclusive as different datasets and horizons slight performance differences. On average across the datasets both the forecasting accuracy and computational performance favors the linear method, with which we conducted the main experiments of this work with this technique.

Table C.4: Empirical evaluation of long multi-horizon multivariate forecasts for NHITS with different interpolation configurations. All other hyperparameters were kept constant across all datasets. MAE and MSE for predictions averaged over eight seeds, the best result is highlighted in bold (lower is better). Percentage difference relative to n. neighbor in the last panel, average across datasets.

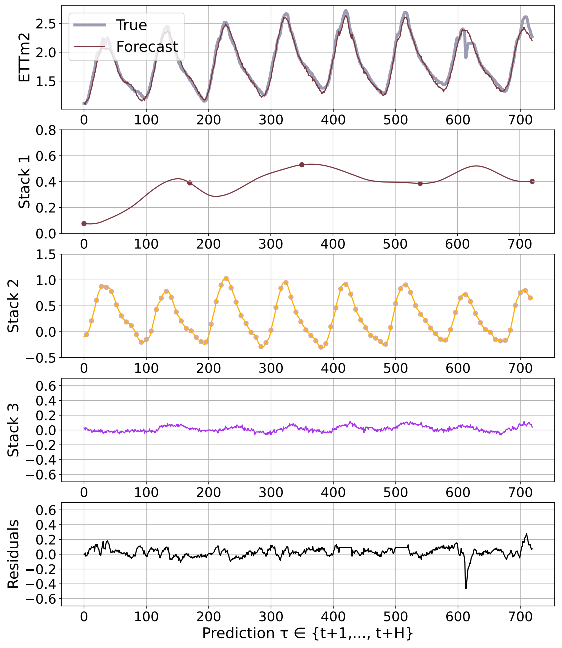
		Linear		Cubic		N.Neighbor	
		MSE	MAE	MSE	MAE	MSE	MAE
$ETTm_2$	96	0.185	0.265	**0.179**	**0.256**	0.180	0.259
	192	0.244	0.308	**0.241**	**0.303**	0.252	0.315
	336	**0.301**	**0.347**	0.314	0.358	0.302	0.351
	720	**0.429**	**0.438**	0.439	0.450	0.442	0.455
ECL	96	0.152	0.257	**0.149**	**0.252**	0.151	0.255
	192	**0.172**	**0.275**	0.174	0.279	0.175	0.279
	336	0.197	0.304	**0.190**	**0.295**	0.211	0.318
	720	**0.248**	**0.347**	0.256	0.353	0.263	0.358
Exchange	96	**0.109**	**0.232**	0.1307	0.254	0.126	0.248
	192	0.280	0.375	**0.247**	**0.357**	0.357	0.416
	336	**0.472**	**0.504**	0.625	0.560	0.646	0.560
	720	**1.241**	**0.823**	1.539	0.925	1.740	0.973
Traffic-L	96	0.405	0.286	**0.402**	**0.282**	0.405	0.359
	192	0.421	0.297	**0.417**	**0.295**	0.419	0.201
	336	0.448	0.318	**0.446**	**0.315**	0.445	0.253
	720	0.527	0.362	0.540	0.366	**0.525**	**0.318**
Weather	96	0.164	0.199	0.162	0.203	**0.161**	0.360
	192	0.224	0.255	0.225	0.257	**0.218**	0.928
	336	**0.285**	**0.311**	0.285	0.304	0.298	0.988
	720	**0.366**	**0.359**	0.380	0.369	0.368	1.047
P. Diff.	96	**-0.907**	**-0.717**	0.146	1.61	0.000	0.000
	192	-5.582	-3.259.	**-7.985**	**-4.332**	0.000	0.000
	336	**-10.516**	**-4.199**	-2.108	-1.455	0.000	0.000
	720	**-15.800**	**-7.042**	-5.480	-1.579	0.000	0.000



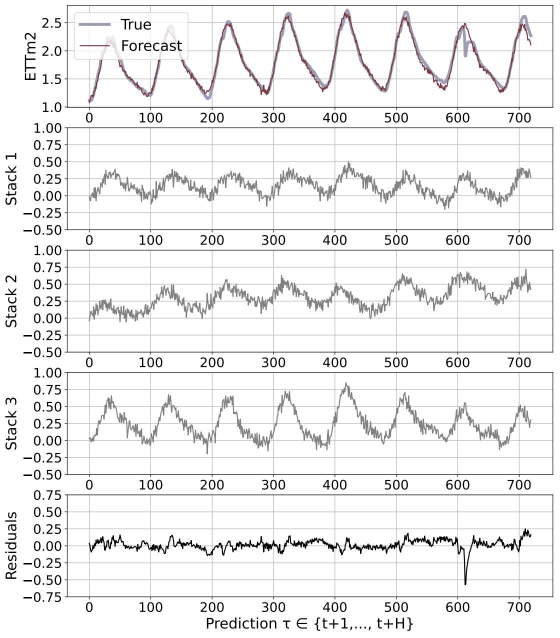
(a) Nearest Neighbors



(b) Linear



(c) Cubic



(d) No Interpolation

Figure C.2: ETTm2 and 720 ahead forecasts using NHITS with different interpolation techniques. The top row shows the original signal and the forecast. The second, third and fourth rows show the forecast components across stack, residuals in the last row.

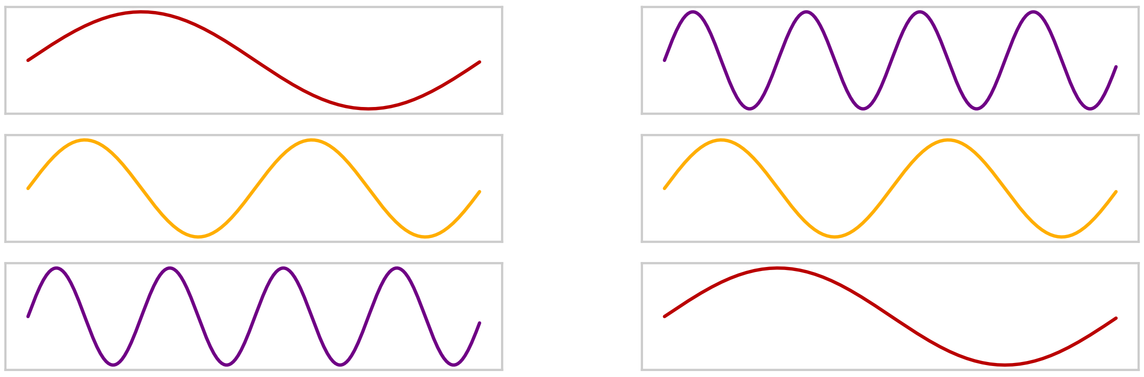

(a) Top-Down **(b)** Bottom-Up

Figure C.3: Hierarchical representation configurations.

C.5.3 Order of Hierarchical Representations

Deep Learning in classic tasks like computer vision and natural language processing is known to learn hierarchical representations from raw data that increase complexity as the information flows through the network. This automatic feature extraction phenomenon is believed to drive to a large degree the algorithms' success (Bengio et al., 2012). Our approach differs from the conventions in the sense that we use a *Top-Down* hierarchy where we prioritize in the synthesis of the predictions to low frequencies and sequentially complement them with higher frequencies details, as explained in Section 6.4. We achieve this with NHITS' *expressiveness ratio* schedules. Our intuition is that the *Top-Down* hierarchy acts as a regularizer and helps the model to focus on the broader factors driving the predictions rather than narrowing its focus at the beginning on the details that compose them. To test these intuitions, we designed an experiment where we inverted the expressiveness ratio schedule into *Bottom-Up* hierarchy predictions and compared the validation performance.

Remarkably, as shown in Table C.5, the *Top-Down* predictions consistently outperform the *Bottom-Up* counterpart. Relative improvements in MAE are 4.6%, in MSE of 7.5%, across horizons and datasets. Our observations match the forecasting community practice that addresses long-horizon predictions by first modeling the long-term seasonal components and then its residuals. Research on long-horizon forecasting has primarily focused on long-term seasonal component, as it is a common belief that it is the most important.

Table C.5: Empirical evaluation of long multi-horizon multivariate forecasts for NHITS with different hierarchical orders. All other hyperparameters were kept constant across all datasets. MAE and MSE for predictions averaged over eight seeds, the best result is highlighted in bold (lower is better). Average percentage difference relative to ascending hierarchy in the last panel.

		Top-Down		Bottom-Up	
		MSE	MAE	MSE	MAE
ETTm$_2$	96	**0.185**	**0.265**	0.191	0.266
	192	**0.244**	**0.308**	0.261	0.320
	336	**0.301**	**0.347**	0.302	0.353
	720	**0.429**	**0.438**	0.440	0.454
ECL	96	**0.152**	**0.257**	0.164	0.270
	192	**0.172**	**0.275**	0.186	0.292
	336	**0.197**	**0.304**	0.217	0.327
	720	**0.248**	**0.347**	0.273	0.369
Exchange	96	**0.109**	**0.232**	0.114	0.242
	192	**0.280**	**0.375**	0.436	0.452
	336	**0.472**	**0.504**	0.654	0.574
	720	**1.241**	**0.823**	1.312	0.861
Traffic-L	96	**0.405**	**0.286**	0.410	0.292
	192	**0.421**	**0.297**	0.427	0.305
	336	**0.448**	**0.318**	0.456	0.323
	720	**0.527**	**0.362**	0.557	0.379
Weather	96	0.164	**0.199**	**0.163**	0.200
	192	0.224	0.255	**0.219**	**0.252**
	336	**0.285**	**0.311**	0.288	0.311
	720	0.366	0.359	**0.365**	**0.355**
P. Diff.	96	**-2.523**	**-2.497**	0.000	0.000
	192	**-12.296**	**-6.793**	0.000	0.000
	336	**-11.176**	**-5.507**	0.000	0.000
	720	**-4.638**	**-3.699**	0.000	0.000

C.6 Multi-rate sampling and Hierarchical Interpolation beyond NHITS

Empirical observations let us infer that the advantages of the NHITS architecture are rooted in its multi-rate hierarchical nature, as both the multi-rate sampling and the hierarchical interpolation complement the long-horizon forecasting task in MLP-based architectures. In this ablation experiment, we quantitatively explore the effects and complementarity of the techniques in an RNN-based architecture.

This experiment follows the Table 6.4 ablation study, reporting the average performance across ETTm2, ECL, Exchange, Traffic-L, and Weather datasets. We define the following set of alternative models: $\texttt{DilRNN}_1$, our proposed model with both multi-rate sampling and hierarchical interpolation, $\texttt{DilRNN}_2$ only hierarchical interpolation, $\texttt{DilRNN}_3$ only multi-rate sampling, DilRNN with no multi-rate sampling or interpolation (corresponds to the original DilRNN (Chang et al., 2017)).

Table C.6 shows that the hierarchical interpolation technique drives the main improvements ($\texttt{DilRNN}_2$), while the combination of both proposed components (hierarchical interpolation and multi-rate sampling) sometimes results in the best performance ($\texttt{DilRNN}_1$), the difference is marginal. Contrary to the clear complementary observed in Table 6.4, the DilRNN does not improve substantially from the multi-rate sampling techniques. We find an explanation in the behavior of the RNN that summarizes past inputs and the current observation of the series, and not the complete whole past data like the MLP-based architectures.

A key takeaway of this experiment is that NHITS' hierarchical interpolation technique exhibits significant benefits in other architectures. Despite these promising results, we decided not to pursue more complex architectures in our work as we found that the interpretability of the NHITS predictions and signal decomposition capabilities was not worth losing.

Table C.6: Empirical evaluation of long multi-horizon multivariate forecasts for DilRNN with/without enhancements. Average MAE and MSE for five datasets, the best result is highlighted in bold, second best in blue (lower is better).

		$\texttt{DilRNN}_1$	$\texttt{DilRNN}_2$	$\texttt{DilRNN}_3$	DilRNN
A. MSE	96	0.346	**0.331**	0.369	0.347
	192	0.539	**0.528**	0.545	0.561
	336	**0.647**	0.691	0.705	0.723
	720	0.765	**0.762**	0.789	0.800
A. MAE	96	**0.343**	0.335	0.352	0.347
	192	0.460	**0.444**	0.462	0.468
	336	**0.513**	0.537	0.539	0.649
	720	0.585	**0.566**	0.600	0.598

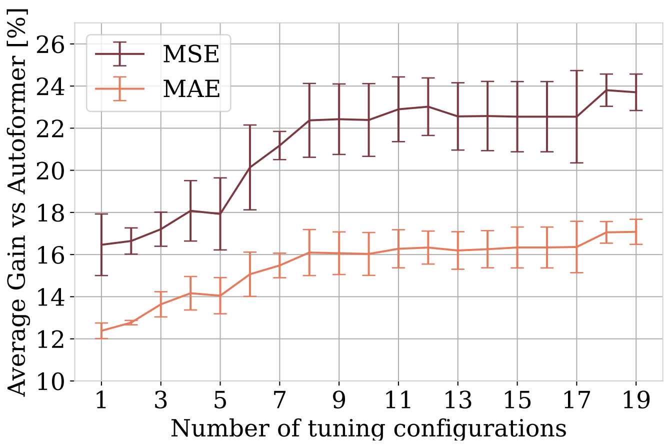


Figure C.4: NHITS performance improvement over Autoformer as a function of explored hyperparameter configurations.

C.7 Hyperparameter Optimization Resources

Computational efficiency has implications for the prediction's accuracy and the cost of deployment. Since forecasting systems are constantly retrained to address distributional shifts, orders-of-magnitude improvements in speed can easily translate into orders-of-magnitude price differences deploying the models. This section explores the implications of computational efficiency in the accuracy gains associated with hyperparameter optimization and training economic costs.

Hyperparameter Optimization. Despite all the progress improving the computation efficiency of Transformer-based methods, see Figure 6.6, their speed and memory requirements make exploring their hyperparameter space unaffordable in practice, considering the amount of GPU computation they still require.

For this experiment we report the iterations of the *hyperparameter optimization phase*, described in Section 6.5.3, where we explore the hyperparameters from Table 6.2 using HYPEROPT, a Bayesian hyperparameter optimization library (Bergstra et al., 2011). As shown in Figure C.4 the exploration exhibits monotonic relative performance gains of NHITS versus the best reported Autoformer (Wu et al., 2021) in the ablation datasets.

Training Economic Costs. We measure the train time for NHITS, NBEATS-G and Transformer-based models, on the six main experiment datasets and 8 runs. We rely on a AWS g4dn.2xlarge, with an NVIDIA T4 16GB GPU.

We differentiate between NHITS$_1$, our method with a single HYPEROPT iteration randomly sampled from Table 6.2, and NHITS$_{20}$ to our method after 20 HYPEROPT iterations. For the Transformer-based models we used optimal hyperparameters as reported in their repositories. Table C.7 shows the measured train time for the models, NHITS$_1$ takes 1.5 hours while more expensive architectures like Autoformer or Informer take 92.6 and 62.1 hours each. Based on hourly prices from January 2022 for the g4dn.2xlarge instance, USD 0.75, the main results of the paper would cost nearly USD 70.0 with `Autoformer`, USD 46.5 with `Informer`, while $\texttt{NHITS}_1$ results can be executed under USD 1.5 and $\texttt{NHITS}_{20}$ with USD 22.8. Figure C.4, shows that $\texttt{NHITS}_1$ achieves a 17% MSE average performance gain over `Autoformer` with 1.6% of a single run cost, and $\texttt{NHITS}_{20}$ almost 25% gain with 33% of a single run cost. A single run does not consider hyperparameter optimization.

$$\text{ExptPrice} = \text{GPUPrice} \times \text{HyperOptIters} \times \text{TrainTime} \times \text{Runs}$$

Table C.7: Train time hours on a `g4dn.2xlarge` instance.

	Horizon	$\texttt{NHITS}_1$	$\texttt{NHITS}_{20}$	Autoformer	Informer	NBEATS-G
A. Time	96/24	0.183	3.66	12.156	9.11	0.291
	192/36	0.257	5.14	16.734	11.598	0.462
	336/48	0.398	7.96	22.73	15.237	0.674
	720/60	0.682	13.64	40.987	26.173	1.249
Total		1.523	30.46	92.607	62.118	2.676

Bibliography

[1] Christoph Heinrich Diedrich Buys-Ballot. *Les changements périodiques de température, dépendants de la nature du soleil et de la lune, mis en rapport avec le pronostic du temps, déduits d'observations néerlandaises de 1729 à 1846.* Kemink, 1847 (cit. on p. 23).

[2] Augustin Cauchy et al. "Méthode générale pour la résolution des systemes d'équations simultanées". In: *Comp. Rend. Sci. Paris* 25.1847 (1847), pp. 536–538 (cit. on p. 17).

[3] Frederick Macaulay. "The Smoothing of Time Series". In: *Journal of the Institute of Actuaries* 62.1 (1931), pp. 181–182. DOI: 10.1017/S0020268100010003. URL: https://www.nber.org/books-and-chapters/smoothing-time-series (cit. on pp. 23, 39).

[4] Herbert Robbins and Sutton Monro. "A stochastic approximation method". In: *The annals of mathematical statistics* (1951), pp. 400–407 (cit. on p. 17).

[5] Charles C Holt. "Forecasting seasonals and trends by exponentially weighted moving averages". In: *(O.N.R. Memorandum No. 52)* (1957). URL: https://doi.org/10.1016/j.ijforecast.2003.09.015 (cit. on pp. 18, 20).

[6] Robert Goodell Brown. "Statistical forecasting for inventory control". In: (1959) (cit. on p. 20).

[7] Peter R. Winters. "Forecasting Sales by Exponentially Weighted Moving Averages". In: *Management Science* 6.3 (1960), pp. 324–342. DOI: 10.1287/mnsc.6.3.324. eprint: https://doi.org/10.1287/mnsc.6.3.324. URL: https://doi.org/10.1287/mnsc.6.3.324 (cit. on p. 20).

[8] Frank Rosenblatt. *Principles of neurodynamics. perceptrons and the theory of brain mechanisms.* Tech. rep. Cornell Aeronautical Lab Inc Buffalo NY, 1961 (cit. on pp. 28, 36, 37, 63).

[9] John A Nelder and Roger Mead. "A simplex method for function minimization". In: *The computer journal* 7.4 (1965), pp. 308–313 (cit. on p. 20).

[10] Julius Shishkin, AH Young, and JC Musgrave. "The X-II variant of the Census II method seasonal adjustment program". In: *Bureau of the Census* Technical Report, No. 15 (1967). URL: https://www.census.gov/library/working-papers/1967/adrm/shiskin-01.html (cit. on p. 39).

[11] Guy H. Orcutt, Harold W. Watts, and John B. Edwards. "Data Aggregation and Information Loss". In: *The American Economic Review* 58.4 (1968), pp. 773–787. ISSN: 00028282. URL: http://www.jstor.org/stable/1815532 (cit. on pp. 55, 61, 71).

[12] J. M. Bates and C. W. J. Granger. "The Combination of Forecasts". In: *OR* 20.4 (1969), pp. 451–468. ISSN: 14732858. URL: http://www.jstor.org/stable/3008764 (cit. on p. 18).

[13] C Carl Pegels. "Exponential forecasting: Some new variations". In: *Management Science* (1969), pp. 311–315 (cit. on p. 20).

[14] Gregory C. Chow and An-loh Lin. "Best Linear Unbiased Interpolation, Distribution, and Extrapolation of Time Series by Related Series". In: *The Review of Economics and Statistics* 53.4 (1971), pp. 372–375. ISSN: 00346535, 15309142. URL: http://www.jstor.org/stable/1928739 (cit. on p. 76).

[15] John D Croston. "Forecasting and stock control for intermittent demands". In: *Journal of the Operational Research Society* 23.3 (1972), pp. 289–303 (cit. on p. 24).

[16] J. A. Nelder and R. W. M. Wedderburn. "Generalized Linear Models". In: *Journal of the Royal Statistical Society. Series A (General)* 135.3 (1972), pp. 370–384. ISSN: 00359238. URL: http://www.jstor.org/stable/2344614 (visited on 06/26/2022) (cit. on pp. 24, 71).

[17] Box George, Jenkins Gwilym, and Reinsel Gregory. *Time Series Analysis: Forecasting and Control*. Prentice Hall, 1976, pp. 799–805 (cit. on p. 21).

[18] James E. Matheson and Robert L. Winkler. "Scoring Rules for Continuous Probability Distributions". In: *Management Science* 22.10 (1976), pp. 1087–1096. ISSN: 00251909, 15265501. URL: http://www.jstor.org/stable/2629907 (cit. on pp. 12, 68).

[19] Roger Koenker and Gilbert Bassett. "Regression Quantiles". In: *Econometrica* 46.1 (1978), pp. 33–50. ISSN: 00129682, 14680262. URL: http://www.jstor.org/stable/1913643 (cit. on p. 15).

[20] Estela Bee Dagum. "The X-II-ARIMA seasonal adjustment method". In: *Statistics Canada. Statistical Sciences* 679 (1980). URL: https://www.census.gov/library/working-papers/1980/adrm/dagum-01.html (cit. on p. 39).

[21] Angus Deaton and John Muellbauer. "An almost ideal demand system". In: *The American economic review* 70.3 (1980), pp. 312–326 (cit. on p. 24).

[22] Christopher A. Sims. "Macroeconomics and Reality". In: *Econometrica* 48.1 (1980), pp. 1–48. ISSN: 00129682, 14680262. URL: http://www.jstor.org/stable/1912017 (visited on 03/06/2023) (cit. on p. 24).

[23] Roque B Fernandez. "A Methodological Note on the Estimation of Time Series". In: *The Review of Economics and Statistics* 63.3 (Aug. 1981), pp. 471–476. URL: https://ideas.repec.org/a/tpr/restat/v63y1981i3p471-76.html (cit. on p. 76).

[24] Robert F Engle. "Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation". In: *Econometrica: Journal of the econometric society* (1982), pp. 987–1007. URL: https://www.jstor.org/stable/1912773 (cit. on p. 22).

[25] Yurii E Nesterov. "A method for solving the convex programming problem with convergence rate O (1/kˆ 2)". In: *Dokl. akad. nauk Sssr*. Vol. 269. 1983, pp. 543–547 (cit. on p. 17).

[26] A. P. Dawid. "Present Position and Potential Developments: Some Personal Views: Statistical Theory: The Prequential Approach". In: *Journal of the Royal Statistical Society. Series A (General)* 147.2 (1984), pp. 278–292. ISSN: 00359238. URL: http://www.jstor.org/stable/2981683 (cit. on p. 10).

[27] Maurice CK Tweedie et al. "An index which distinguishes between some important exponential families". In: *Statistics: Applications and new directions: Proc. Indian statistical institute golden Jubilee International conference*. Vol. 579. 1984, pp. 579–604 (cit. on p. 24).

[28] Everette S Gardner Jr. "Exponential smoothing: The state of the art". In: *Journal of forecasting* 4.1 (1985), pp. 1–28. URL: https://onlinelibrary.wiley.com/doi/10.1002/for.3980040103 (cit. on p. 20).

[29] Tim Bollerslev. "Generalized autoregressive conditional heteroskedasticity". In: *Journal of econometrics* 31.3 (1986), pp. 307–327. URL: https://www.sciencedirect.com/science/article/pii/0304407686900631 (cit. on p. 22).

[30] Nigel Meade and J. Armstrong. "Review: Long Range Forecasting: From Crystal Ball to Computer (2nd Edition)". In: *Journal of the Operational Research Society* 37 (May 1986), pp. 533–535. DOI: 10.2307/2582679 (cit. on p. 11).

[31] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. "Learning representations by back-propagating errors". In: *nature* 323.6088 (1986), pp. 533–536 (cit. on p. 29).

[32] Bent Jorgensen. "Exponential Dispersion Models". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 49.2 (1987), pp. 127–162. ISSN: 00359246. URL: http://www.jstor.org/stable/2345415 (visited on 03/07/2023) (cit. on p. 24).

[33] B. G. Lindsay. "Composite likelihood methods". In: *Contemporary Mathematics* 80 (1988), pp. 221–239 (cit. on pp. 16, 60).

[34] George Cybenko. "Approximation by superpositions of a sigmoidal function". In: *Mathematics of control, signals and systems* 2.4 (1989), pp. 303–314 (cit. on p. 27).

[35] James D Hamilton. "A new approach to the economic analysis of nonstationary time series and the business cycle". In: *Econometrica: Journal of the econometric society* (1989), pp. 357–384 (cit. on p. 23).

[36] Yann LeCun et al. "Generalization and network design strategies". In: *Connectionism in perspective* 19.143-155 (1989), p. 18 (cit. on p. 30).

[37] Robert B. Cleveland, William S. Cleveland, Jean E. McRae, and Irma Terpenning. "STL: A Seasonal-Trend Decomposition Procedure Based on Loess (with Discussion)". In: *Journal of Official Statistics* 6 (1990), pp. 3–73 (cit. on pp. 23, 39).

[38] Jeffrey L. Elman. "Finding structure in time". In: *Cognitive Science* 14.2 (1990), pp. 179–211. URL: https://onlinelibrary.wiley.com/doi/abs/10.1207/s15516709cog1402_1 (cit. on p. 29).

[39] Charles W. Gross and Jeffrey E. Sohl. "Disaggregation methods to expedite product line forecasting". In: *Journal of Forecasting* 9.3 (1990), pp. 233–254. DOI: 10.1002/for.3980090304. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/for.3980090304 (cit. on p. 55).

[40] Adrian R Pagan and G William Schwert. "Alternative models for conditional stock volatility". In: *Journal of econometrics* 45.1-2 (1990), pp. 267–290 (cit. on p. 23).

[41] Kurt Hornik. "Approximation capabilities of multilayer feedforward networks". In: *Neural Networks* 4.2 (1991), pp. 251–257. ISSN: 0893-6080. DOI: https://doi.org/10.1016/0893-6080(91)90009-T. URL: https://www.sciencedirect.com/science/article/pii/089360809190009T (cit. on pp. 27, 109).

[42] Ingrid Daubechies. *Ten lectures on wavelets*. SIAM, 1992 (cit. on pp. 80, 87).

[43] Diane Lambert. “Zero-Inflated Poisson Regression, With an Application to Defects in Manufacturing”. In: *Technometrics* 34.1 (1992), pp. 1–14. DOI: 10.1080/00401706.1992.10485228. eprint: https://www.tandfonline.com/doi/pdf/10.1080/00401706.1992.10485228. URL: https://www.tandfonline.com/doi/abs/10.1080/00401706.1992.10485228 (cit. on pp. 24, 107).

[44] Andrew R Barron. “Universal approximation bounds for superpositions of a sigmoidal function”. In: *IEEE Transactions on Information theory* 39.3 (1993), pp. 930–945 (cit. on p. 110).

[45] Christopher M. Bishop. *Mixture density networks.* Tech. rep. Birmingham: Aston University, 1994. URL: https://publications.aston.ac.uk/id/eprint/373/ (cit. on p. 53).

[46] Corinna Cortes and Vladimir Vapnik. “Support-Vector Networks”. In: *Mach. Learn.* 20.3 (Sept. 1995), pp. 273–297. ISSN: 0885-6125. DOI: 10.1023/A:1022627411411. URL: http://dx.doi.org/10.1023/A:1022627411411 (cit. on p. 25).

[47] Leo Breiman. “Bagging Predictors”. In: *Machine Learning* 24.2 (Aug. 1996), pp. 123–140. ISSN: 0885-6125. DOI: 10.1023/A:1018054314350. URL: https://doi.org/10.1023/A:1018054314350 (cit. on pp. 26, 48).

[48] Yoav Freund and Robert E Schapire. “A decision-theoretic generalization of on-line learning and an application to boosting”. In: *Journal of computer and system sciences* 55.1 (1997), pp. 119–139 (cit. on p. 26).

[49] Vladimir Vapnik, Harris Drucker, Christopher J. C. Burges, Linda Kaufman, and Smola Alex. “Support Vector Regression Machines”. In: *Advances in Neural Information Processing Systems* 1.9 (1997), pp. 155–161 (cit. on p. 25).

[50] David F. Findley, Brian C. Monsell, William R. Bell, Mark C. Otto, and Bor-Chung Chen. “New Capabilities and Methods of the X-12-ARIMA Seasonal-Adjustment Program”. In: *Journal of Business & Economic Statistics* 16.2 (1998), pp. 127–152. ISSN: 07350015. URL: http://www.jstor.org/stable/1392565 (cit. on p. 39).

[51] Yann LeCun, Leon Bottou, Genevieve B. Orr, and Klaus -Robert Müller. “Efficient Back-Prop”. In: *Neural Networks: Tricks of the Trade.* Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 9–50. ISBN: 978-3-540-49430-0. DOI: 10.1007/3-540-49430-8_2. URL: https://doi.org/10.1007/3-540-49430-8_2 (cit. on p. 41).

[52] Gene Fliedner. “An Investigation of Aggregate Variable Time Series Forecast Strategies with Specific Subaggregate Time Series Statistical Correlation”. In: *Computers and Operations Research* 26.10–11 (Sept. 1999), pp. 1133–1149. ISSN: 0305-0548. DOI: 10.1016/S0305-0548(99)00017-9. URL: https://doi.org/10.1016/S0305-0548(99)00017-9 (cit. on p. 55).

[53] Chang-Jin Kim, Charles R Nelson, et al. “State-space models with regime switching: classical and Gibbs-sampling approaches with applications”. In: *MIT Press Books* 1 (1999) (cit. on p. 23).

[54] Vladimir Vapnik. *The Nature of Statistical Learning Theory.* Springer Science & Business Media, 1999 (cit. on pp. 15, 25).

[55] V. Assimakopoulos and K. Nikolopoulos. “The theta model: a decomposition approach to forecasting”. In: *International Journal of Forecasting* 16.4 (2000). The M3-Competition, pp. 521–530. ISSN: 0169-2070. DOI: https://doi.org/10.1016/S0169-2070(00)00066-2. URL: https://www.sciencedirect.com/science/article/pii/S0169207000000662 (cit. on p. 22).

[56] Felix A. Gers, Fred Cummins, and Jürgen Schmidhuber. “Learning to forget: continual prediction with LSTM”. English. In: *Neural Computation* 12 (2000), pp. 2451–2471. URL: https://digital-library.theiet.org/content/conferences/10.1049/cp_19991218 (cit. on p. 29).

[57] Spyros Makridakis and Michèle Hibon. “The M3-Competition: results, conclusions and implications”. In: *International Journal of Forecasting* 16.4 (2000). The M3- Competition, pp. 451–476. ISSN: 0169-2070. DOI: https://doi.org/10.1016/S0169-2070(00)00057-1. URL: https://www.sciencedirect.com/science/article/pii/S0169207000000571 (cit. on p. 22).

[58] Theodore B Trafalis and Huseyin Ince. “Support vector machine for regression and applications to financial forecasting”. In: *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium.* Vol. 6. IEEE. 2000, pp. 348–353 (cit. on p. 25).

[59] Leo Breiman. “Random Forests”. In: *Machine Learning* 45 (2001), pp. 5–32. DOI: 10.1023/A:1010933404324. URL: https://doi.org/10.1023/A:1010933404324 (cit. on p. 26).

[60] Christopher K. I. Williams and Matthias Seeger. “Using the Nyström Method to Speed Up Kernel Machines”. In: *Advances in Neural Information Processing Systems 13*. Ed. by T. K. Leen, T. G. Dietterich, and V. Tresp. MIT Press, 2001, pp. 682–688. URL: http://papers.nips.cc/paper/1866-using-the-nystrom-method-to-speed-up-kernel-machines.pdf (cit. on p. 25).

[61] Francis Diebold and Roberto Mariano. “Comparing Predictive Accuracy”. In: *Journal of Business & Economic Statistics* 20 (Feb. 2002), pp. 134–44. DOI: 10.1080/07350015.1995.10524599. URL: https://www.sas.upenn.edu/~fdiebold/papers/paper68/pa.dm.pdf (cit. on pp. 13, 45).

[62] E. Meijering. “A chronology of interpolation: from ancient astronomy to modern signal and image processing”. In: *Proceedings of the IEEE* 90.3 (2002), pp. 319–342. DOI: 10.1109/5.993400. URL: https://ieeexplore.ieee.org/document/993400 (cit. on p. 76).

[63] Rob J. Hyndman and Baki Billah. “Unmasking the Theta method”. In: *International Journal of Forecasting* 19.2 (2003), pp. 287–290. ISSN: 0169-2070. DOI: https://doi.org/10.1016/S0169-2070(01)00143-1. URL: https://www.sciencedirect.com/science/article/pii/S0169207001001431 (cit. on p. 22).

[64] James W Taylor. “Exponential smoothing with a damped multiplicative trend”. In: *International journal of Forecasting* 19.4 (2003), pp. 715–725 (cit. on p. 20).

[65] Ben S Bernanke, Jean Boivin, and Piotr Eliasz. “Measuring the effects of monetary policy: a factor-augmented vector autoregressive (FAVAR) approach”. In: *The Quarterly journal of economics* 120.1 (2005), pp. 387–422 (cit. on p. 24).

[66] Reid Basher. “Global early warning systems for natural hazards: Systematic and people-centred”. In: *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences* 364 (Sept. 2006), pp. 2167–82. DOI: 10.1098/rsta.2006.1819 (cit. on p. 73).

[67] Luc Bauwens, Sébastien Laurent, and Jeroen VK Rombouts. “Multivariate GARCH models: a survey”. In: *Journal of applied econometrics* 21.1 (2006), pp. 79–109 (cit. on p. 24).

[68] Raffaella Giacomini and Halbert White. “Tests of Conditional Predictive Ability”. In: *Econometrica* 74.6 (2006), pp. 1545–1578. DOI: https://doi.org/10.1111/j.1468-0262.2006.00718.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1468-0262.2006.00718.x (cit. on pp. 13, 45).

[69] Rob J. Hyndman and Anne B. Koehler. “Another look at measures of forecast accuracy”. In: *International Journal of Forecasting* 22.4 (2006), pp. 679–688. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2006.03.001. URL: http://www.sciencedirect.com/science/article/pii/S0169207006000239 (cit. on pp. 11, 45, 68).

[70] Massimiliano Marcellino, James H. Stock, and Mark W. Watson. “A comparison of direct and iterated multistep AR methods for forecasting macroeconomic time series”. In: *Journal of Econometrics* 135.1 (2006), pp. 499–526. ISSN: 0304-4076. DOI: https://doi.org/10.1016/j.jeconom.2005.07.020. URL: https://www.sciencedirect.com/science/article/pii/S030440760500165X (cit. on p. 74).

[71] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*. Vol. 2. 3. MIT press Cambridge, MA, 2006 (cit. on p. 26).

[72] Eric Ghysels, Arthur Sinko, and Rossen Valkanov. “MIDAS Regressions: Further Results and New Directions”. In: *Econometric Reviews* 26.1 (2007), pp. 53–90. DOI: 10.1080/07474930600972467. URL: https://doi.org/10.1080/07474930600972467 (cit. on p. 76).

[73] F. Laio and S. Tamea. “Verification tools for probabilistic forecasts of continuous hydrological variables”. In: *Hydrology and Earth System Sciences* 11.4 (2007), pp. 1267–1277 (cit. on p. 12).

[74] Yuan Yao, Lorenzo Rosasco, and Caponneto Andrea. “On Early Stopping in Gradient Descent Learning”. In: *Constructive Approximation* 26(2) (2007), pp. 289–315. URL: https://doi.org/10.1007/s00365-006-0663-2 (cit. on pp. 46, 69).

[75] Rob Hyndman, Anne B Koehler, J Keith Ord, and Ralph D Snyder. *Forecasting with exponential smoothing: the state space approach*. Springer Science & Business Media, 2008 (cit. on pp. 18, 20).

[76] Rob J. Hyndman and Yeasmin Khandakar. “Automatic Time Series Forecasting: The forecast Package for R”. In: *Journal of Statistical Software, Articles* 27.3 (2008), pp. 1–22. ISSN: 1548-7660. DOI: 10.18637/jss.v027.i03. URL: https://www.jstatsoft.org/v027/i03 (cit. on pp. 18, 21, 71, 84, 113, 114).

[77] Ali Rahimi and Benjamin Recht. “Random Features for Large-Scale Kernel Machines”. In: *Advances in Neural Information Processing Systems 20*. Ed. by J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis. Curran Associates, Inc., 2008, pp. 1177–1184. URL: http://papers.nips.cc/paper/3182-random-features-for-large-scale-kernel-machines.pdf (cit. on p. 25).

[78] Pavel Senin. “Dynamic time warping algorithm review”. In: *Information and Computer Science Department University of Hawaii at Manoa Honolulu, USA* 855.1-23 (2008), p. 40 (cit. on p. 25).

[79] Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Vol. 2. Springer, 2009 (cit. on p. 15).

[80] Rob J Hyndman and Shu Fan. “Density forecasting for long-term peak electricity demand”. In: *IEEE Transactions on Power Systems* 25.2 (2009), pp. 1142–1153 (cit. on p. 73).

[81] Nicholas I Sapankevych and Ravi Sankar. “Time series prediction using support vector machines: a survey”. In: *IEEE computational intelligence magazine* 4.2 (2009), pp. 24–38 (cit. on p. 25).

[82] Sylvain Arlot and Alain Celisse. “A survey of cross-validation procedures for model selection”. In: *Statistics Surveys* 4.none (Jan. 2010). ISSN: 1935-7516. DOI: 10.1214/09-ss054. URL: http://dx.doi.org/10.1214/09-SS054 (cit. on p. 14).

[83] Michelle T. Armesto, Kristie M. Engemann, and Michael T. Owyang. “Forecasting with Mixed Frequencies”. In: *Federal Reserve Bank of St. Louis Review* 92 (2010), pp. 521–536. URL: https://research.stlouisfed.org/publications/review/2010/11/01/forecasting-with-mixed-frequencies (cit. on p. 76).

[84] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. “Algorithms for Hyper-Parameter Optimization”. In: *Advances in Neural Information Processing Systems*. Ed. by J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Q. Weinberger. Vol. 24. Curran Associates, Inc., 2011, pp. 2546–2554. URL: https://proceedings.neurips.cc/paper/2011/file/86e8f7ab32cfd12577bc2619bc635690-Paper.pdf (cit. on pp. 18, 46, 69, 83, 104, 106, 122).

[85] Tilmann Gneiting and Roopesh Ranjan. “Comparing density forecasts using threshold- and quantile-weighted scoring rules”. In: *Journal of Business & Economic Statistics* 29.3 (2011), pp. 411–422 (cit. on pp. 12, 68).

[86] Rob J. Hyndman, Roman A. Ahmed, George Athanasopoulos, and Han Lin Shang. “Optimal combination forecasts for hierarchical time series”. In: *Computational Statistics & Data Analysis* 55.9 (2011), pp. 2579–2589. ISSN: 0167-9473. DOI: https://doi.org/10.1016/j.csda.2011.03.006. URL: http://www.sciencedirect.com/science/article/pii/S0167947311000971 (cit. on pp. 52, 55).

[87] Alysha M. De Livera, Rob J. Hyndman, and Ralph D. Snyder. “Forecasting Time Series With Complex Seasonal Patterns Using Exponential Smoothing”. In: *Journal of the American Statistical Association* 106.496 (2011), pp. 1513–1527. DOI: 10.1198/jasa.2011.tm09771. eprint: https://doi.org/10.1198/jasa.2011.tm09771. URL: https://doi.org/10.1198/jasa.2011.tm09771 (cit. on p. 39).

[88] Konstantinos Nikolopoulos, Aris Syntetos, J E Boylan, Fotios Petropoulos, and Vassilios Assimakopoulos. In: *Journal of the Operational Research Society* 62.3 (2011), pp. 544–554. ISSN: 0160-5682. DOI: 10.1057/jors.2010.32 (cit. on p. 24).

[89] Cristiano Varin, Nancy Reid, and David Firth. “AN OVERVIEW OF COMPOSITE LIKELIHOOD METHODS”. In: *Statistica Sinica* 21.1 (2011), pp. 5–42. ISSN: 10170405, 19968507. URL: http://www.jstor.org/stable/24309261 (cit. on pp. 16, 60).

[90] Yoshua Bengio, Aaron C. Courville, and Pascal Vincent. “Unsupervised Feature Learning and Deep Learning: A Review and New Perspectives”. In: *CoRR* abs/1206.5538 (2012). arXiv: 1206.5538. URL: http://arxiv.org/abs/1206.5538 (cit. on p. 119).

[91] James Durbin and Siem Jan Koopman. *Time series analysis by state space methods*. Vol. 38. OUP Oxford, 2012 (cit. on p. 24).

[92] Christopher B Field, Vicente Barros, Thomas F Stocker, and Qin Dahe. *Managing the risks of extreme events and disasters to advance climate change adaptation: special report of the intergovernmental panel on climate change*. Cambridge University Press, 2012 (cit. on p. 73).

[93] Kevin P Murphy. *Machine learning: a probabilistic perspective*. Cambridge, MA, 2012 (cit. on p. 16).

[94] Alex Graves. “Generating Sequences With Recurrent Neural Networks”. In: *Computing Research Repository* abs/1308.0850 (2013). arXiv: 1308.0850. URL: http://arxiv.org/abs/1308.0850 (cit. on p. 31).

[95] S. Roberts, M. Osborne, M. Ebden, S. Reece, N. Gibson, and S. Aigrain. “Gaussian processes for time-series modelling”. In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 371.1984 (2013), p. 20110550. DOI: 10.1098/rsta.2011.0550. eprint: https://royalsocietypublishing.org/doi/pdf/10.1098/rsta.2011.0550. URL: https://royalsocietypublishing.org/doi/abs/10.1098/rsta.2011.0550 (cit. on p. 26).

[96] U.S. Census Bureau. *X-13-ARIMA-SEATS Reference Manual*. U.S. Census Bureau. Washington, DC, 2013. URL: http://www.census.gov/ts/x13as/docX13AS.pdf (cit. on p. 39).

[97] Yukun Bao, Tao Xiong, and Zhongyi Hu. “Multi-step-ahead time series prediction using multiple-output support vector regression”. In: *Neurocomputing* 129 (2014), pp. 482–493. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2013.09.010. URL: https://www.sciencedirect.com/science/article/pii/S092523121300917X (cit. on p. 76).

[98] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. “Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* abs/1406.1078 (2014), pp. 1724–1734. arXiv: 1406.1078. URL: http://arxiv.org/abs/1406.1078 (cit. on pp. 29, 63).

[99] Jan Chorowski, Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. “End-to-end continuous speech recognition using attention-based recurrent nn: First results”. In: *arXiv preprint arXiv:1412.1602* (2014) (cit. on p. 30).

[100] Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. “Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling”. In: *NIPS 2014* Workshop on Deep Learning (2014). arXiv: 1412.3555. URL: http://arxiv.org/abs/1412.3555 (cit. on p. 29).

[101] Rob J Hyndman, Alan Lee, and Earo Wang. *Fast computation of reconciled forecasts for hierarchical and grouped time series*. Monash Econometrics and Business Statistics Working Papers 17/14. Monash University, Department of Econometrics and Business Statistics, Dec. 2014. URL: https://ideas.repec.org/p/msh/ebswps/2014-17.html (cit. on p. 52).

[102] Diederik P. Kingma and Jimmy Ba. *ADAM: A Method for Stochastic Optimization*. cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations (ICLR), San Diego, 2015. 2014. URL: http://arxiv.org/abs/1412.6980 (cit. on pp. 17, 46, 69, 84).

[103] Jakub Nowotarski, Eran Raviv, Stefan Trück, and Rafał Weron. “An empirical comparison of alternative schemes for combining electricity spot price forecasts”. In: *Energy Economics* 46.C (2014), pp. 395–412. DOI: 10.1016/j.eneco.2014.07.0. URL: https://ideas.repec.org/a/eee/eneeco/v46y2014icp395-412.html (cit. on p. 48).

[104] Hasim Sak, Andrew W. Senior, and Françoise Beaufays. “Long Short-Term Memory Based Recurrent Neural Network Architectures for Large Vocabulary Speech Recognition”.

In: *Computing Research Repository* abs/1402.1128 (2014). arXiv: 1402.1128. URL: http://arxiv.org/abs/1402.1128 (cit. on p. 29).

[105] J.W Taylor. “Short-Term Electricity Demand Forecasting Using Double Seasonal Exponential Smoothing”. In: *Journal of Operational Research Society* 54 (2014), pp. 799–805 (cit. on p. 23).

[106] Rafał Weron. “Electricity price forecasting: A review of the state-of-the-art with a look into the future”. In: *International Journal of Forecasting* 30.4 (2014), pp. 1030–1081. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2014.08.008. URL: https://www.sciencedirect.com/science/article/pii/S0169207014001083 (cit. on pp. 34, 48).

[107] Albert Boggess and Francis J Narcowich. *A first course in wavelets with Fourier analysis.* John Wiley & Sons, 2015 (cit. on p. 109).

[108] G.E.P. Box, G.M. Jenkins, G.C. Reinsel, and G.M. Ljung. *Time Series Analysis: Forecasting and Control.* Wiley Series in Probability and Statistics. Wiley, 2015. ISBN: 9781118674925. URL: https://books.google.com/books?id=rNt5CgAAQBAJ (cit. on p. 18).

[109] George Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control.* John Wiley & Sons, 2015 (cit. on p. 21).

[110] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. “Fast and accurate deep network learning by exponential linear units (elus)”. In: *arXiv preprint arXiv:1511.07289* (2015) (cit. on p. 69).

[111] Alexander Dokumentov and Rob J. Hyndman. *STR: A Seasonal-Trend Decomposition Procedure Based on Regression.* Monash Econometrics and Business Statistics Working Papers 13/15. Monash University, Department of Econometrics and Business Statistics, 2015. URL: https://ideas.repec.org/p/msh/ebswps/2015-13.html (cit. on pp. 23, 39).

[112] Siu Kwan Lam, Antoine Pitrou, and Stanley Seibert. “Numba: A LLVM-Based Python JIT Compiler”. In: *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC.* LLVM ’15. Austin, Texas: Association for Computing Machinery, 2015. ISBN: 9781450340052. DOI: 10.1145/2833157.2833162. URL: https://doi.org/10.1145/2833157.2833162 (cit. on p. 6).

[113] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep Learning”. In: *Nature* 521.7553 (2015), pp. 436–444 (cit. on p. 1).

[114] Tianqi Chen et al. “MXNet: A Flexible and Efficient Machine Learning Library for Heterogeneous Distributed Systems”. In: *CoRR* abs/1512.01274 (2015). URL: http://arxiv.org/abs/1512.01274 (cit. on pp. 6, 69).

[115] Atiya Amir and Ben Taieb Souhaib. “A Bias and Variance Analysis for Multistep-Ahead Time Series Forecasting”. In: *IEEE transactions on neural networks and learning systems* 27.1 (2016), pp. 2162–2388. URL: https://pubmed.ncbi.nlm.nih.gov/25807572/ (cit. on pp. 63, 74, 76).

[116] Lucia Parisio Angelica Gianfreda and Matteo Pelagatti. “The Impact of RES in the Italian Day–Ahead and Balancing Markets”. In: *The Energy Journal* 37 (2016), pp. 161–184. DOI: 10.5547/01956574.37.SI2.agia. URL: http://www.iaee.org/en/publications/ejarticle.aspx?id=2736 (cit. on p. 34).

[117] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. “Neural Machine Translation by Jointly Learning to Align and Translate”. In: *3rd International Conference on Learning*

Representations, ICLR 2015. 2016. arXiv: 1409.0473 [cs.CL]. URL: https://arxiv.org/abs/1409.0473 (cit. on p. 30).

[118] Tianqi Chen and Carlos Guestrin. "XGBoost: A Scalable Tree Boosting System". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. San Francisco, California, USA: Association for Computing Machinery, 2016, pp. 785–794. ISBN: 9781450342322. DOI: 10.1145/2939672.2939785. URL: https://doi.org/10.1145/2939672.2939785 (cit. on p. 26).

[119] Matthew M Churpek, Richa Adhikari, and Dana P Edelson. "The value of vital sign trends for detecting clinical deterioration on the wards". In: *Resuscitation* 102 (2016), pp. 1–5 (cit. on p. 73).

[120] Jose A. Fiorucci, Tiago R. Pellegrini, Francisco Louzada, Fotios Petropoulos, and Anne B. Koehler. "Models for optimising the theta method and their relationship to state space models". In: *International Journal of Forecasting* 32.4 (2016), pp. 1151–1161. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2016.02.005. URL: https://www.sciencedirect.com/science/article/pii/S0169207016300243 (cit. on p. 22).

[121] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016 (cit. on p. 27).

[122] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior, and Koray Kavukcuoglu. "WaveNet: A Generative Model for Raw Audio". In: *Computer Research Repository* abs/1609.03499 (2016). arXiv: 1609.03499. URL: http://arxiv.org/abs/1609.03499 (cit. on pp. 30, 40, 63).

[123] Ivan Svetunkov. *Complex exponential smoothing*. Lancaster University (United Kingdom), 2016 (cit. on p. 20).

[124] Bartosz Uniejewski, Jakub Nowotarski, and Rafał Weron. "Automated Variable Selection and Shrinkage for Day-Ahead Electricity Price Forecasting". In: *Energies* 9.8 (2016). ISSN: 1996-1073. URL: https://www.mdpi.com/1996-1073/9/8/621 (cit. on p. 48).

[125] George Athanasopoulos, Rob J Hyndman, Nikolaos Kourentzes, and Fotios Petropoulos. "Forecasting with temporal hierarchies". In: *European Journal of Operational Research* 262.1 (2017), pp. 60–74 (cit. on p. 52).

[126] Souhaib Ben Taieb, James W. Taylor, and Rob J. Hyndman. "Coherent Probabilistic Forecasts for Hierarchical Time Series". In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, Aug. 2017, pp. 3348–3357. URL: http://proceedings.mlr.press/v70/taieb17a.html (cit. on pp. 52, 55, 56).

[127] Joos-Hendrik Böse, Valentin Flunkert, Jan Gasthaus, Tim Januschowski, Dustin Lange, David Salinas, Sebastian Schelter, Matthias Seeger, and Yuyang Wang. "Probabilistic Demand Forecasting at Scale". In: *Proc. VLDB Endow.* 10.12 (Aug. 2017), pp. 1694–1705. ISSN: 2150-8097. DOI: 10.14778/3137765.3137775. URL: https://doi.org/10.14778/3137765.3137775 (cit. on p. 57).

[128] Shiyu Chang, Yang Zhang, Wei Han, Mo Yu, Xiaoxiao Guo, Wei Tan, Xiaodong Cui, Michael Witbrock, Mark A Hasegawa-Johnson, and Thomas S Huang. "Dilated Recurrent Neural Networks". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30.

Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper/2017/file/32bb90e8976aab5298d5da10fe66f21d-Paper.pdf (cit. on pp. 30, 84, 121).

[129] Dheeru Dua and Casey Graff. *UCI Machine Learning Repository*. 2017. URL: http://archive.ics.uci.edu/ml (cit. on p. 65).

[130] Boris Hanin and Mark Sellke. *Approximating Continuous Functions by ReLU Nets of Minimal Width*. 2017. URL: https://arxiv.org/abs/1710.11278 (cit. on p. 110).

[131] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. "LightGBM: A highly efficient gradient boosting decision tree". In: *Advances in neural information processing systems* 30 (2017) (cit. on p. 26).

[132] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. "Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks". In: *CoRR* abs/1703.07015 (2017). arXiv: 1703.07015. URL: http://arxiv.org/abs/1703.07015 (cit. on p. 81).

[133] Nikolay Laptev, Jason Yosinsk, Li Li Erran, and Slawek Smyl. "Time-series extreme event forecasting with neural networks at UBER". In: *34th International Conference on Machine Learning ICML 2017, Time Series Workshop*. 2017. URL: http://www.cs.columbia.edu/~lierranli/publications/TSW2017_paper.pdf (cit. on p. 76).

[134] Han Lin Shang and Rob J. Hyndman. "Grouped Functional Time Series Forecasting: An Application to Age-Specific Mortality Rates". In: *Journal of Computational and Graphical Statistics* 26.2 (2017), pp. 330–343. DOI: 10.1080/10618600.2016.1237877. eprint: https://doi.org/10.1080/10618600.2016.1237877. URL: https://doi.org/10.1080/10618600.2016.1237877 (cit. on p. 56).

[135] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need". In: *Advances in neural information processing systems* 30 (2017) (cit. on p. 30).

[136] Long Wang, Zijun Zhang, and Jieqiu Chen. "Short-term electricity price forecasting with stacked denoising autoencoders". In: *IEEE Transactions on Power Systems* 32.4 (2017), pp. 2673–2681. DOI: 10.1109/TPWRS.2016.2628873 (cit. on p. 35).

[137] Ruofeng Wen, Kari Torkkola, Balakrishnan Narayanaswamy, and Dhruv Madeka. "A Multi-Horizon Quantile Recurrent Forecaster". In: *31st Conference on Neural Information Processing Systems NIPS 2017, Time Series Workshop*. 2017. arXiv: 1711.11053 [stat.ML]. URL: https://arxiv.org/abs/1711.11053 (cit. on pp. 12, 27, 31, 53, 57, 62, 74, 76).

[138] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling". In: *Computing Research Repository* abs/1803.01271 (2018). arXiv: 1803.01271. URL: http://arxiv.org/abs/1803.01271 (cit. on pp. 30, 40).

[139] H. Chitsaz, P. Zamani-Dehkordi, H. Zareipour, and P.P. Parikh. "Electricity price forecasting for operational scheduling of behind-the-meter storage systems". In: *IEEE Transactions on Smart Grid* 9.6 (2018), pp. 6612–6622. DOI: 10.1109/TSG.2017.2717282 (cit. on p. 34).

[140] Corporación Favorita. *Corporación Favorita Grocery Sales Forecasting*. Kaggle Competition. 2018. URL: https://www.kaggle.com/c/favorita-grocery-sales-forecasting/ (cit. on p. 65).

[141] Massimo Guidolin and Manuela Pedio. *Essentials of time series for financial applications*. Academic Press, 2018 (cit. on p. 23).

[142] Katarzyna Hubicka, Grzegorz Marcjasz, and Rafal Weron. *A note on averaging day-ahead electricity price forecasts across calibration windows*. HSC Research Reports HSC/18/03. Hugo Steinhaus Center, Wroclaw University of Technology, July 2018. URL: https://ideas.repec.org/p/wuu/wpaper/hsc1803.html (cit. on p. 48).

[143] Rob J Hyndman and George Athanasopoulos. *Forecasting: Principles and Practice*. available at https://otexts.com/fpp2/. Melbourne, Australia: OTexts, 2018 (cit. on p. 55).

[144] Rob J Hyndman and George Athanasopoulos. *Forecasting: principles and practice*. OTexts, 2018 (cit. on pp. 11, 14, 20).

[145] Jesus Lago, Fjo De Ridder, and Bart De Schutter. "Forecasting spot electricity prices: Deep learning approaches and empirical comparison of traditional algorithms". In: *Applied Energy* 221 (2018), pp. 386–405. ISSN: 0306-2619. DOI: https://doi.org/10.1016/j.apenergy.2018.02.069. URL: http://www.sciencedirect.com/science/article/pii/S030626191830196X (cit. on pp. 35, 46, 48).

[146] Jesus Lago, Fjo De Ridder, Peter Vrancx, and Bart De Schutter. "Forecasting day-ahead electricity prices in Europe: The importance of considering market integration". In: *Applied Energy* 211 (2018), pp. 890–903. ISSN: 0306-2619. DOI: https://doi.org/10.1016/j.apenergy.2017.11.098. URL: https://www.sciencedirect.com/science/article/pii/S0306261917316999 (cit. on p. 41).

[147] Dhruv Madeka, Lucas Swiniarski, Dean Foster, Leo Razoumov, Kari Torkkola, and Ruofeng Wen. "Sample path generation for probabilistic demand forecasting". In: *ICML workshop on Theoretical Foundations and Applications of Deep Generative Models*. 2018 (cit. on p. 57).

[148] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. "Statistical and Machine Learning forecasting methods: Concerns and ways forward". In: *PLoS One* 13(3) (2018), e0194889. URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0194889 (cit. on p. 19).

[149] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. "Statistical and Machine Learning forecasting methods: Concerns and ways forward". In: *PLOS ONE* 13.3 (Mar. 2018), pp. 1–26. DOI: 10.1371/journal.pone.0194889. URL: https://doi.org/10.1371/journal.pone.0194889 (cit. on p. 55).

[150] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. "The M4 Competition: Results, findings, conclusion and way forward". In: *International Journal of Forecasting* 34.4 (2018), pp. 802–808. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2018.06.001. URL: http://www.sciencedirect.com/science/article/pii/S0169207018300785 (cit. on pp. 31, 52, 57).

[151] K. Mayer and S. Trück. "Electricity markets around the world". In: *Journal of Commodity Markets* 9 (2018), pp. 77–100. URL: https://doi.org/10.1016/j.jcomm.2018.02.001 (cit. on p. 34).

[152] Mehrdad Setayesh Nazar, Ashkan Eslami Fard, Alireza Heidari, Miadreza Shafie-khah, and João P.S. Catalão. "Hybrid model using three-stage algorithm for simultaneous load and price forecasting". In: *Electric Power Systems Research* 165 (2018), pp. 214–228. DOI: 10.1016/j.epsr.2018.09.004 (cit. on p. 35).

[153] Jakub Nowotarski and Rafał Weron. "Recent advances in electricity price forecasting: A review of probabilistic forecasting". In: *Renewable and Sustainable Energy Reviews* 81 (2018), pp. 1548–1568. ISSN: 1364-0321. DOI: https://doi.org/10.1016/j.rser.2017.05.234 (cit. on p. 34).

[154] Gamakumara Puwasala, G. Panagiotelis Anastasios Athanasopoulos, and Rob J Hyndman. "Probabilisitic Forecasts in Hierarchical Time Series". In: *Department of Econometrics and Business Statistics Working Paper Series 11/18* (2018) (cit. on p. 56).

[155] Syama Sundar Rangapuram, Matthias W Seeger, Jan Gasthaus, Lorenzo Stella, Yuyang Wang, and Tim Januschowski. "Deep State Space Models for Time Series Forecasting". In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett. Vol. 31. Curran Associates, Inc., 2018. URL: https://proceedings.neurips.cc/paper/2018/file/5cf68969fb67aa6082363a6d4e6468e2-Paper.pdf (cit. on p. 70).

[156] Sean J Taylor and Benjamin Letham. "Forecasting at scale". In: *The American Statistician* 72.1 (2018), pp. 37–45 (cit. on pp. 23, 114).

[157] B. Uniejewski, R. Weron, and F. Ziel. "Variance Stabilizing Transformations for Electricity Spot Price Forecasting". In: *IEEE Transactions on Power Systems* 33.2 (2018), pp. 2219–2229. DOI: 10.1109/TPWRS.2017.2734563 (cit. on p. 41).

[158] Bing Yu, Haoteng Yin, and Zhanxing Zhu. "Spatio-temporal Graph Convolutional Neural Network: A Deep Learning Framework for Traffic Forecasting". In: *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI)*. 2018. URL: http://arxiv.org/abs/1709.04875 (cit. on p. 57).

[159] F. Ziel and R. Steinert. "Probabilistic mid- and long-term electricity price forecasting". In: *Renewable and Sustainable Energy Reviews* 94 (2018), pp. 251–266. URL: https://arxiv.org/abs/1703.10806 (cit. on p. 34).

[160] F. Ziel and R. Steinert. "Probabilistic mid- and long-term electricity price forecasting". In: *Renewable and Sustainable Energy Reviews* 94 (2018), pp. 251–266. URL: https://arxiv.org/abs/1703.10806 (cit. on p. 73).

[161] Ahmed M. Alaa and Mihaela van der Schaar. "Attentive State-Space Modeling of Disease Progression". In: *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. Vol. 32. Curran Associates, Inc., 2019. URL: https://proceedings.neurips.cc/paper/2019/file/1d0932d7f57ce74d9d9931a2c6db8a06-Paper.pdf (cit. on p. 73).

[162] David Bolin and Jonas Wallin. *Local scale invariance and robustness of proper scoring rules*. 2019. DOI: 10.48550/ARXIV.1912.05642. URL: https://arxiv.org/abs/1912.05642 (cit. on p. 68).

[163] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. "Generating Long Sequences with Sparse Transformers". In: *CoRR* abs/1904.10509 (2019). arXiv: 1904.10509. URL: http://arxiv.org/abs/1904.10509 (cit. on p. 74).

[164] William Falcon et al. "PyTorch Lightning". In: *GitHub. Note: https://github.com/PyTorchLightning/pytorch lightning* 3 (2019) (cit. on p. 6).

[165] Chenyou Fan, Yuze Zhang, Yi Pan, Xiaoyue Li, Chi Zhang, Rong Yuan, Di Wu, Wensheng Wang, Jian Pei, and Heng Huang. "Multi-Horizon Time Series Forecasting with Temporal

Attention Learning". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. KDD '19. Anchorage, AK, USA: Association for Computing Machinery, 2019, pp. 2527–2535. ISBN: 9781450362016. DOI: 10.1145/3292500.3330662. URL: https://doi.org/10.1145/3292500.3330662 (cit. on p. 73).

[166] Jan Gasthaus, Konstantinos Benidis, Bernie Wang, Syama Sundar Rangapuram, David Salinas, Valentin Flunkert, and Tim Januschowski. "Probabilistic Forecasting with Spline Quantile Function RNNs". In: *AISTATS*. 2019 (cit. on p. 76).

[167] Jooyoung Jeon, Anastasios Panagiotelis, and Fotios Petropoulos. "Probabilistic forecast reconciliation with applications to wind power and electric load". In: *European Journal of Operational Research* 279.2 (2019), pp. 364–379. ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2019.05.020. URL: https://www.sciencedirect.com/science/article/pii/S0377221719304242 (cit. on p. 56).

[168] Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. "Enhancing the Locality and Breaking the Memory Bottleneck of Transformer on Time Series Forecasting". In: *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. Vol. 32. Curran Associates, Inc., 2019. URL: http://arxiv.org/abs/1907.00235 (cit. on pp. 73, 74, 76, 81, 84, 113, 114).

[169] Francisco Martínez, María Pilar Frías, María Dolores Pérez, and Antonio Jesús Rivera. "A methodology for applying k-nearest neighbor to time series forecasting". In: *Artificial Intelligence Review* 52.3 (2019), pp. 2019–2037 (cit. on p. 25).

[170] Kostas I Nikolopoulos and Dimitrios D Thomakos. *Forecasting with the theta method: theory and applications*. John Wiley & Sons, 2019 (cit. on p. 22).

[171] Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alché-Buc, E. Fox, and R. Garnett. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf (cit. on pp. 6, 46, 84).

[172] Yulia Rubanova, Ricky T. Q. Chen, and David Duvenaud. "Latent ODEs for Irregularly-Sampled Time Series". In: *Advances in Neural Information Processing Systems 33 (NeurIPS 2019)*. 2019. URL: http://arxiv.org/abs/1907.03907 (cit. on p. 76).

[173] Satya Narayan Shukla and Benjamin M. Marlin. *Interpolation-Prediction Networks for Irregularly Sampled Time Series*. cite arxiv:1412.6980Comment: Published as a conference paper at the 7th International Conference for Learning Representations (ICLR), New Orleans, 2019. 2019. URL: http://arxiv.org/abs/1412.6980 (cit. on p. 76).

[174] Slawek Smyl. "A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting". In: *International Journal of Forecasting* (July 2019). DOI: 10.1016/j.ijforecast.2019.03.017 (cit. on pp. 27, 31, 33, 48).

[175] Ben Taieb Souhaib and Koo Bonsoo. "Regularized Regression for Hierarchical Forecasting Without Unbiasedness Conditions". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. KDD '19. Anchorage, AK, USA: Association for Computing Machinery, 2019, pp. 1337–1347. ISBN: 9781450362016. DOI:

10.1145/3292500.3330976. URL: https://doi.org/10.1145/3292500.3330976 (cit. on pp. 55, 64, 65, 71, 103).

[176] Tourism Australia, Canberra. *Detailed Tourism Research Australia (2005), Travel by Australians.* Accessed at https://robjhyndman.com/publications/hierarchical-tourism/. Sept. 2019 (cit. on p. 65).

[177] Shanika L. Wickramasuriya, George Athanasopoulos, and Rob J. Hyndman. “Optimal Forecast Reconciliation for Hierarchical and Grouped Time Series Through Trace Minimization”. In: *Journal of the American Statistical Association* 114.526 (2019), pp. 804–819. DOI: 10.1080/01621459.2018.1448825. URL: https://robjhyndman.com/publications/mint/ (cit. on pp. 52, 55, 64, 71, 103).

[178] S. Zhou, L. Zhou, M. Mao, H. Tai, and Y. Wan. “An Optimized Heterogeneous Structure LSTM Network for Electricity Price Forecasting”. In: *IEEE Access* 7 (2019), pp. 108161–108173. DOI: 10.1109/ACCESS.2019.2932999 (cit. on p. 81).

[179] Alexander Alexandrov, Konstantinos Benidis, Michael Bohlke-Schneider, Valentin Flunkert, Jan Gasthaus, Tim Januschowski, Danielle C. Maddix, Syama Rangapuram, David Salinas, Jasper Schulz, Lorenzo Stella, Ali Caner Tarkmen, and Yuyang Wang. “GluonTS: Probabilistic and Neural Time Series Modeling in Python”. In: *Journal of Machine Learning Research* 21.116 (2020), pp. 1–6. URL: http://jmlr.org/papers/v21/19-820.html (cit. on p. 6).

[180] Amir F. Atiya. “Why does forecast combination work so well?” In: *International Journal of Forecasting* 36.1 (2020). M4 Competition, pp. 197–200. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2019.03.010. URL: https://www.sciencedirect.com/science/article/pii/S0169207019300779 (cit. on p. 48).

[181] Konstantinos Benidis, Syama Sundar Rangapuram, Valentin Flunkert, Bernie Wang, Danielle Maddix, Caner Turkmen, Jan Gasthaus, Michael Bohlke-Schneider, David Salinas, Lorenzo Stella, Laurent Callot, and Tim Januschowski. “Neural forecasting: Introduction and literature overview”. In: *Computing Research Repository* (2020). URL: http://arxiv.org/abs/2004.10240 (cit. on p. 76).

[182] Jose Fiorucci and Francisco Louzada. *forecTheta: Forecasting Time Series by Theta Models.* . R package version 2.6.2. 2020. URL: https://cran.r-project.org/web/packages/forecTheta/index.html (cit. on p. 22).

[183] Angelica Gianfreda, Francesco Ravazzolo, and Luca Rossini. “Comparing the forecasting performances of linear models for electricity prices with high RES penetration”. In: *International Journal of Forecasting* 36.3 (2020), pp. 974–986. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2019.11.002. URL: https://www.sciencedirect.com/science/article/pii/S0169207019302596 (cit. on p. 34).

[184] Rob J Hyndman, Alan Lee, Earo Wang, Shanika Wickramasuriya, and Maintainer Earo Wang. *Package hts: Hierarchical and Grouped Time Series.* . R package version 6.0.1. 2020. URL: https://CRAN.R-project.org/package=hts (cit. on p. 6).

[185] Tim Januschowski, Jan Gasthaus, Yuyang Wang, David Salinas, Valentin Flunkert, Michael Bohlke-Schneider, and Laurent Callot. “Criteria for classifying forecasting methods”. In: *International Journal of Forecasting* 36.1 (2020). M4 Competition, pp. 167–177. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2019.05.008. URL: https://www.sciencedirect.com/science/article/pii/S0169207019301529 (cit. on p. 19).

[186] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. "Reformer: The Efficient Transformer". In: *8th International Conference on Learning Representations, (ICLR 2020).* 2020. URL: https://arxiv.org/abs/2001.04451 (cit. on pp. 74, 76, 84, 113, 114).

[187] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. "The M4 Competition: 100,000 time series and 61 forecasting methods". In: *International Journal of Forecasting* 36.1 (2020). M4 Competition, pp. 54–74. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2019.04.014. URL: https://www.sciencedirect.com/science/article/pii/S0169207019301128 (cit. on pp. 1, 26, 33, 76).

[188] Spyros Makridakis, Evangelos Spiliotis, and Vassilis Assimakopoulos. "The M5 Accuracy competition: Results, findings and conclusions". In: *International Journal of Forecasting* (Oct. 2020). URL: https://www.researchgate.net/publication/344487258_The_M5_Accuracy_competition_Results_findings_and_conclusions (cit. on pp. 31, 57).

[189] G. Marcjasz. "Forecasting electricity prices using deep neural networks: A robust hyper-parameter selection scheme". In: *Energies* 13.18 (2020), p. 13184605 (cit. on p. 35).

[190] Peru Muniain and Florian Ziel. "Probabilistic forecasting in day-ahead electricity markets: Simulating peak and off-peak prices". In: *International Journal of Forecasting* 36.4 (2020), pp. 1193–1210. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2019.11.006. URL: https://www.sciencedirect.com/science/article/pii/S0169207019302675 (cit. on p. 34).

[191] M. Narajewski and F. Ziel. "Econometric modelling and forecasting of intraday electricity prices". In: *Journal of Commodity Markets* 19 (2020), p. 100107. DOI: 10.1016/j.jcomm.2019.100107 (cit. on p. 34).

[192] Boris N. Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. "N-BEATS: Neural basis expansion analysis for interpretable time series forecasting". In: *8th International Conference on Learning Representations, ICLR 2020.* 2020. URL: https://openreview.net/forum?id=r1ecqn4YwB (cit. on pp. 3, 31, 33, 40, 48, 73, 74, 77, 86, 114).

[193] Anastasios Panagiotelis, Puwasala Gamakumara, George Athanasopoulos, and Rob J Hyndman. *Probabilistic Forecast Reconciliation: Properties, Evaluation and Score Optimisation.* Monash Econometrics and Business Statistics Working Papers 26/20. Monash University, Department of Econometrics and Business Statistics, Dec. 2020. URL: https://ideas.repec.org/p/msh/ebswps/2020-26.html (cit. on pp. 52, 55).

[194] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. "DeepAR: Probabilistic forecasting with autoregressive recurrent networks". In: *International Journal of Forecasting* 36.3 (2020), pp. 1181–1191. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2019.07.001. URL: https://www.sciencedirect.com/science/article/pii/S0169207019301888 (cit. on pp. 31, 114).

[195] Evangelos Spiliotis, Vassilios Assimakopoulos, and Spyros Makridakis. "Generalizing the Theta method for automatic forecasting". In: *European Journal of Operational Research* 284.2 (2020), pp. 550–558. ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2020.01.007. URL: https://www.sciencedirect.com/science/article/pii/S0377221720300242 (cit. on p. 22).

[196] Evangelos Spiliotis, Fotios Petropoulos, Nikolaos Kourentzes, and Vassilios Assimakopoulos. "Cross-temporal aggregation: Improving the forecast accuracy of hierarchical electricity consumption". In: *Applied Energy* 261 (2020), p. 114339 (cit. on p. 52).

[197] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. "Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting". In: *The Association for the Advancement of Artificial Intelligence Conference 2021 (AAAI 2021).* abs/2012.07436 (2020). arXiv: 2012.07436. URL: https://arxiv.org/abs/2012.07436 (cit. on pp. 30, 74, 76, 81, 83, 84, 113, 114).

[198] Kasun Bandara, Rob J Hyndman, and Christoph Bergmeir. "MSTL: a seasonal-trend decomposition algorithm for time series with multiple seasonal patterns". In: *arXiv preprint arXiv:2107.13462* (2021). URL: https://arxiv.org/pdf/2107.13462.pdf (cit. on p. 23).

[199] Boris Banushev and Richard Barclay. *Enhancing trading strategies through cloud services and machine learning*. Jan. 2021. URL: https://aws.amazon.com/blogs/industries (cit. on p. 76).

[200] Souhaib Ben Taieb, James W. Taylor, and Rob J. Hyndman. "Hierarchical Probabilistic Forecasting of Electricity Demand With Smart Meter Data". In: *Journal of the American Statistical Association* 116.533 (2021), pp. 27–43. DOI: 10.1080/01621459.2020.1736081. eprint: https://doi.org/10.1080/01621459.2020.1736081. URL: https://doi.org/10.1080/01621459.2020.1736081 (cit. on pp. 52, 56).

[201] Casper Solheim Bojer and Jens Peder Meldgaard. "Kaggle forecasting competitions: An overlooked learning opportunity". In: *International Journal of Forecasting* 37.2 (2021), pp. 587–603 (cit. on p. 26).

[202] Cristian Challu, Kin G. Olivares, Gus Welter, and Artur Dubrawski. "DMIDAS: Deep Mixed Data Sampling Regression for Long Multi-Horizon Time Series Forecasting". In: *9th International Conference of Machine Learning, (ICML 2021)* Workshop on Time Series (2021). URL: https://arxiv.org/abs/2106.05860 (cit. on p. 5).

[203] Carson Eisenach, Yagna Patel, and Dhruv Madeka. "MQTransformer: Multi-Horizon Forecasts with Context Dependent and Feedback-Aware Attention". In: *Submitted to Proceedings of the 38th International Conference on Machine Learning*. Ed. by Maria Florina Balcan and Marina Meila. PMLR. Working Paper version available at arXiv:2009.14799, Aug. 2021 (cit. on pp. 31, 53, 57, 62, 72).

[204] Xing Han, Sambarta Dasgupta, and Joydeep Ghosh. "Simultaneously Reconciled Quantile Forecasting of Hierarchically Related Time Series". In: *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics.* Ed. by Arindam Banerjee and Kenji Fukumizu. Vol. 130. Proceedings of Machine Learning Research. PMLR, Apr. 2021, pp. 190–198. URL: http://proceedings.mlr.press/v130/han21a.html (cit. on pp. 53, 57).

[205] Jesus Lago, Grzegorz Marcjasz, Bart De Schutter, and Rafał Weron. "Forecasting day-ahead electricity prices: A review of state-of-the-art algorithms, best practices and an open-access benchmark". In: *Applied Energy* 293 (2021), p. 116983. ISSN: 0306-2619. DOI: https://doi.org/10.1016/j.apenergy.2021.116983. URL: https://www.sciencedirect.com/science/article/pii/S0306261921004529 (cit. on pp. 11, 34, 35, 45, 48, 49, 95, 96).

[206] Jesus Lago, Grzegorz Marcjasz, Bart De Schutter, and Rafal Weron. *Erratum to 'Forecasting day-ahead electricity prices: A review of state-of-the-art algorithms, best practices and an open-access benchmark' [Appl. Energy 293 (2021) 116983]*. WORking papers in Management Science (WORMS) WORMS/21/12. Department of Operations Research,

Business Intelligence, Wroclaw University of Science, and Technology, July 2021. URL: https://ideas.repec.org/p/ahh/wpaper/worms2112.html (cit. on pp. 49, 96).

[207] W. Li and D.M. Becker. "Day-ahead electricity price prediction applying hybrid models of LSTM-based deep learning methods and feature selection algorithms under consideration of market coupling". In: *Energy* 237 (2021), p. 121543 (cit. on p. 35).

[208] Bryan Lim, Sercan Ö. Arık, Nicolas Loeff, and Tomas Pfister. "Temporal Fusion Transformers for interpretable multi-horizon time series forecasting". In: *International Journal of Forecasting* 37.4 (2021), pp. 1748–1764. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2021.03.012. URL: https://www.sciencedirect.com/science/article/pii/S0169207021000637 (cit. on pp. 31, 73, 74).

[209] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. "Predicting/hypothesizing the findings of the M5 competition". In: *International Journal of Forecasting* (2021). ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2021.09.014. URL: https://www.sciencedirect.com/science/article/pii/S0169207021001631 (cit. on pp. 1, 24, 26, 76).

[210] Kin G. Olivares, Nganba Meetei, Ruijun Ma, Rohan Reddy, Mengfei Cao, and Lee Dicker. "Probabilistic Hierarchical Forecasting with Deep Poisson Mixtures". In: *International Journal of Forecasting, submitted* Accepted Paper version available at arXiv:2110.13179 (2021). URL: https://arxiv.org/abs/2110.13179 (cit. on pp. 5, 11, 76).

[211] Biswajit Paria, Rajat Sen, Amr Ahmed, and Abhimanyu Das. "Hierarchically Regularized Deep Forecasting". In: *Submitted to Proceedings of the 39th International Conference on Machine Learning*. PMLR. Working Paper version available at arXiv:2106.07630, 2021 (cit. on pp. 57, 64, 76).

[212] Syama Sundar Rangapuram, Lucien D. Werner, Konstantinos Benidis, Pedro Mercado, Jan Gasthaus, and Tim Januschowski. "End-to-End Learning of Coherent Probabilistic Forecasts for Hierarchical Time Series". In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by Maria Florina Balcan and Marina Meila. Proceedings of Machine Learning Research. PMLR, Aug. 2021 (cit. on pp. 6, 53, 56, 57, 64, 67, 71, 76, 103).

[213] Suman V. Ravuri, Karel Lenc, Matthew Willson, Dmitry Kangin, Rémi Lam, Piotr Mirowski, Megan Fitzsimons, Maria Athanassiadou, Sheleem Kashem, Sam Madge, Rachel Prudden, Amol Mandhane, Aidan Clark, Andrew Brock, Karen Simonyan, Raia Hadsell, Niall H. Robinson, Ellen Clancy, Alberto Arribas, and Shakir Mohamed. "Skillful Precipitation Nowcasting using Deep Generative Models of Radar". In: *Nature* 597 (2021), pp. 672–691. URL: https://www.nature.com/articles/s41586-021-03854-z.pdf (cit. on p. 57).

[214] Artemios-Anargyros Semenoglou, Evangelos Spiliotis, Spyros Makridakis, and Vassilios Assimakopoulos. "Investigating the accuracy of cross-learning time series forecasting methods". In: *International Journal of Forecasting* 37.3 (2021), pp. 1072–1084. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2020.11.009. URL: https://www.sciencedirect.com/science/article/pii/S0169207020301850 (cit. on pp. 27, 31, 52).

[215] Aris A Syntetos and John E Boylan. *Intermittent demand forecasting: Context, methods and applications*. John Wiley & Sons, 2021 (cit. on p. 24).

[216] Bartosz Uniejewski and Rafał Weron. "Regularized quantile regression averaging for probabilistic electricity price forecasting". In: *Energy Economics* 95 (2021), p. 105121.

ISSN: 0140-9883. DOI: https://doi.org/10.1016/j.eneco.2021.105121. URL: https://www.sciencedirect.com/science/article/pii/S0140988321000268 (cit. on p. 34).

[217] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. "Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting". In: *Advances in Neural Information Processing Systems 35 (NeurIPS 2021)*. Ed. by M. Ranzato, A. Beygelzime, P.S. Liang, J.W. Vaughan, and Y. Dauphin. 2021. URL: https://arxiv.org/abs/2106.13008 (cit. on pp. 30, 74, 76, 81, 83, 84, 86, 113, 114, 122).

[218] Federico Garza, Max Mergenthaler Canseco, Cristian Challú, and Kin G. Olivares. *StatsForecast: Lightning Fast Forecasting with Statistical and Econometric Models*. PyCon Salt Lake City, Utah, US 2022. 2022. URL: https://github.com/Nixtla/statsforecast (cit. on p. 6).

[219] Tim Januschowski, Yuyang Wang, Kari Torkkola, Timo Erkkilä, Hilaf Hasson, and Jan Gasthaus. "Forecasting with trees". In: *International Journal of Forecasting* 38.4 (2022). Special Issue: M5 competition, pp. 1473–1481. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2021.10.004. URL: https://www.sciencedirect.com/science/article/pii/S0169207021001679 (cit. on p. 26).

[220] Harshavardhan Kamarthi, Lingkai Kong, Alexander Rodriguez, Chao Zhang, and B. Prakash. "PROFHIT: Probabilistic Robust Forecasting for Hierarchical Time-series". In: *Computing Research Repository* (June 2022). URL: https://arxiv.org/abs/2206.07940 (cit. on p. 57).

[221] Spyros Makridakis, Evangelos Spiliotis, Vassilios Assimakopoulos, Zhi Chen, Anil Gaba, Ilia Tsetlin, and Robert L. Winkler. "The M5 uncertainty competition: Results, findings and conclusions". In: *International Journal of Forecasting* 38.4 (2022). Special Issue: M5 competition, pp. 1365–1385. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2021.10.009. URL: https://www.sciencedirect.com/science/article/pii/S0169207021001722 (cit. on p. 68).

[222] Kin G. Olivares, Cristian Challu, Grzegorz Marcjasz, Rafał Weron, and Artur Dubrawski. "Neural basis expansion analysis with exogenous variables: Forecasting electricity prices with NBEATSx". In: *International Journal of Forecasting* (2022). ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2022.03.001. URL: https://www.sciencedirect.com/science/article/pii/S0169207022000413 (cit. on pp. 5, 57, 73, 74, 77).

[223] Kin G. Olivares, Cristian Challú, Federico Garza, Max Mergenthaler Canseco, and Artur Dubrawski. *NeuralForecast: User friendly state-of-the-art neural forecasting models*. PyCon Salt Lake City, Utah, US 2022. 2022. URL: https://github.com/Nixtla/neuralforecast (cit. on p. 6).

[224] Kin G. Olivares, Federico Garza, David Luo, Cristian Challú, Max Mergenthaler, Souhaib Ben Taieb, Shanika L. Wickramasuriya, and Artur Dubrawski. "HierarchicalForecast: A Reference Framework for Hierarchical Forecasting in Python". In: *Work in progress paper, submitted to Journal of Machine Learning Research*. abs/2207.03517 (2022). URL: https://arxiv.org/abs/2207.03517 (cit. on pp. 6, 55).

[225] Fotios Petropoulos, Daniele Apiletti, Vassilios Assimakopoulos, Mohamed Zied Babai, Devon K. Barrow, Souhaib Ben Taieb, Christoph Bergmeir, Ricardo J. Bessa, Jakub Bijak, John E. Boylan, Jethro Browell, Claudio Carnevale, Jennifer L. Castle, Pasquale Cirillo, Michael P. Clements, Clara Cordeiro, Fernando Luiz Cyrino Oliveira, Shari De Baets,

Alexander Dokumentov, Joanne Ellison, Piotr Fiszeder, Philip Hans Franses, David T. Frazier, Michael Gilliland, M. Sinan Gönül, Paul Goodwin, Luigi Grossi, Yael Grushka-Cockayne, Mariangela Guidolin, Massimo Guidolin, Ulrich Gunter, Xiaojia Guo, Renato Guseo, Nigel Harvey, David F. Hendry, Ross Hollyman, Tim Januschowski, Jooyoung Jeon, Victor Richmond R. Jose, Yanfei Kang, Anne B. Koehler, Stephan Kolassa, Nikolaos Kourentzes, Sonia Leva, Feng Li, Konstantia Litsiou, Spyros Makridakis, Gael M. Martin, Andrew B. Martinez, Sheik Meeran, Theodore Modis, Konstantinos Nikolopoulos, Dilek Önkal, Alessia Paccagnini, Anastasios Panagiotelis, Ioannis Panapakidis, Jose M. Pavía, Manuela Pedio, Diego J. Pedregal, Pierre Pinson, Patrícia Ramos, David E. Rapach, J. James Reade, Bahman Rostami-Tabar, Michał Rubaszek, Georgios Sermpinis, Han Lin Shang, Evangelos Spiliotis, Aris A. Syntetos, Priyanga Dilini Talagala, Thiyanga S. Talagala, Len Tashman, Dimitrios Thomakos, Thordis Thorarinsdottir, Ezio Todini, Juan Ramón Trapero Arenas, Xiaoqian Wang, Robert L. Winkler, Alisa Yusupova, and Florian Ziel. “Forecasting: theory and practice”. In: *International Journal of Forecasting* (2022). ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2021.11.001. URL: https://www.sciencedirect.com/science/article/pii/S0169207021001758 (cit. on p. 19).

[226] Seuk Wai Phoong, Seuk Yen Phoong, and Shi Ling Khek. “Systematic literature review with bibliometric analysis on Markov switching model: Methods and applications”. In: *SAGE Open* 12.2 (2022), p. 21582440221093062 (cit. on p. 23).

[227] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. “FEDformer: Frequency Enhanced Decomposed Transformer for Long-term Series Forecasting”. In: *Computing Research Repository* abs/2201.12740 (2022). arXiv: 2201.12740. URL: https://arxiv.org/abs/2201.12740 (cit. on pp. 30, 84).

[228] Cristian Challu, Kin G. Olivares, Boris N. Oreshkin, Federico Garza, Max Mergenthaler, and Artur Dubrawski. “N-HiTS: Neural Hierarchical Interpolation for Time Series Forecasting”. In: *The Association for the Advancement of Artificial Intelligence Conference 2023 (AAAI 2023)*. 2023. URL: https://arxiv.org/abs/2201.12886 (cit. on p. 5).

[229] Kin G. Olivares, David Luo, Stefania La Vattiata, Cristian Challu, Max Mergenthaler, and Artur Dubrawski. “HINT: Hierarchical Neural Networks For Coherent Probabilistic Forecasting”. In: *International Conference of Machine Learning* Workshop paper available at arXiv:2110.13179 (2023). URL: https://arxiv.org/abs/2110.13179 (cit. on p. 5).

[230] Anastasios Panagiotelis, Puwasala Gamakumara, George Athanasopoulos, and Rob J. Hyndman. “Probabilistic forecast reconciliation: Properties, evaluation and score optimisation”. In: *European Journal of Operational Research* 306.2 (2023), pp. 693–706. ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2022.07.040. URL: https://www.sciencedirect.com/science/article/pii/S0377221722006087 (cit. on pp. 52, 56).

[231] Shanika L. Wickramasuriya. “Probabilistic forecast reconciliation under the Gaussian framework”. In: *Accepted at Journal of Business and Economic Statistics* (2023) (cit. on pp. 52, 56).