FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Predicting the Performance of Neural Networks using their Weights for Time Series Forecasting

Guilherme Dias da Rocha Pereira

Mestrado em Engenharia Informática e Computação

Supervisor: Vítor Cerqueira

Second Supervisor: Prof. Carlos Soares

July 21, 2025

Predicting the Performance of Neural Networks using their Weights for Time Series Forecasting

Guilherme Dias da Rocha Pereira

Mestrado em Engenharia Informática e Computação

July 21, 2025

Resumo

Redes neuronais alcançaram elevado desempenho na previsão de séries temporais, embora o processo de treino continue computacionalmente exigente. Esta tese explora se as propriedades estatísticas dos pesos de uma rede podem servir como indicadores fiáveis de desempenho, permitindo estimativas sem o treino completo do modelo.

Assim, enquadramos *meta-learning* como uma ferramenta capaz de prever o desempenho das redes neuronais utilizando metadados extraídos a partir das configurações internas dos pesos em vários pontos de controle de treino. Através da captura de propriedades estatísticas e espectrais das matrizes de pesos, construímos modelos preditivos capazes de prever o desempenho final de uma rede.

Multilayer perceptrons foram avaliados utilizando seis conjuntos de dados de séries temporais. Os resultados demonstram que os metadados derivados dos pesos podem efetivamente aproximar a precisão do modelo, particularmente em tarefas de classificação, com fortes correlações observadas entre o desempenho previsto e o desempenho real. A análise da importância das características revelou que diferentes tarefas dependem de aspectos distintos do espaço de pesos, sendo as características espectrais da camada final mais informativas para tarefas de classificação e as normas de camadas iniciais mais relevantes para tarefas de regressão. Estas conclusões permitem compreender melhor a dinâmica de treino das redes neuronais e oferecem uma abordagem prática para melhorar a eficiência da avaliação de modelos na previsão de séries temporais.

Abstract

Neural networks have achieved state-of-the-art performance in time series forecasting, yet their training remains computationally intensive and resource-demanding. This thesis explores whether statistical properties of network weights can serve as reliable indicators of final forecasting accuracy, enabling performance estimation without full model training.

We propose a meta-learning framework that predicts forecasting performance of neural networks using metadata extracted from the internal weight configurations at multiple training checkpoints. We capture both statistical and spectral properties of weight matrices, building predictive models capable of forecasting a network's final accuracy.

Experimental evaluations were conducted using multilayer perceptrons trained on six benchmark time series datasets. Results demonstrate that weight-derived metadata can effectively approximate model accuracy, particularly in ranking tasks, with strong correlations observed between predicted and actual performance. Feature importance analysis revealed that different tasks rely on distinct aspects of the weight space, with output-layer spectral features most informative for classification and lower-layer norms more predictive for regression. These findings advance the understanding of neural network training dynamics and offer a practical approach to improving the efficiency of model evaluation in time series forecasting.

Funding

This work is a result of Agenda “Center for Responsible AI”, nr. C645008882-00000055, investment project nr. 62, financed by the Recovery and Resilience Plan (PRR) and by European Union - NextGeneration EU. Funded by the European Union – NextGenerationEU. Funded by the European Union under the Horizon Europe Framework Programme Grant Agreement Nº: 101095387. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Commission. Neither the European Union nor the European Health and Digital Executive Agency can be held responsible for them; UID/00027 of the LIACC - Artificial Intelligence and Computer Science Laboratory - funded by Fundação para a Ciência e a Tecnologia, I.P./ MCTES through the national funds.

Guilherme Pereira

Agradecimentos

First of all, I would like to thank my family. They are the ones who allowed me to pursue this path and always supported me. Without them, any of this wouldn't be possible. I also want to thank my friends who were part of these incredible five years of my life. It's a period that can feel both short and long at the same time, but it's a phase that I don't think I'll ever forget. Finally, I want to thank both of my supervisors, Vitor Cerqueira and Prof. Carlos Soares, for their amazing insights and wisdom.

Guilherme Dias da Rocha Pereira

Contents

List of Figures

List of Tables

Abbreviations and Symbols

AUC	Area Under the Curve
DNN	Deep Neural Network
GBM	Gradient Boosting Machine
GNN	Graph Neural Network
GRU	Gated Recurrent Units
LOG LOSS	Logarithmic Loss
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
MASE	Mean Absolute Scaled Error
MLP	Multilayer Perceptron
MSE	Mean Squared Error
NAS	Neural Architecture Search
ROC	Receiver Operating Characteristic
RNN	Recurrent Neural Network
sMAPE	Symmetric Mean Absolute Percentage Error
TSF	Time Series Forecasting

Chapter 1

Introduction

1.1 Context and Motivation

The rapid advancement of deep learning has led to significant progress across domains such as natural language processing, computer vision [14], and time series forecasting [8, 23, 43]. Despite these achievements, training neural networks remains computationally intensive, requiring substantial resources in time, energy, and hardware [44]. Variability introduced by hyperparameter tuning, architecture selection, and early training decisions often results in substantial differences in final model performance, thereby increasing development costs and model uncertainty [52].

Furthermore, neural networks are frequently characterized as black-box models due to the opacity of their internal mechanisms [11, 15, 49, 52]. Although performance gains have been substantial, limited understanding of training dynamics impedes interpretability, model debugging, and the design of more efficient learning strategies [32, 44]. This limitation often precludes their wider adoption, especially in sensitive domains such as healthcare.

1.2 Time Series Forecasting

Time series forecasting is an extensively studied topic in the literature [23]. This task involves using historical data to predict the value of future observations. Forecasting is relevant across varied domains such as energy [22, 27], finance, industry [28, 48], or healthcare [35]. Accurate models can be essential to guide decision-making within organizations.

Deep neural network methods are effective approaches for time series forecasting problems. Results from benchmark datasets and competitions such as M4 [28] and M5 [27] show that neural networks can achieve state-of-the-art performance relative to traditional methods such as ARIMA [18]. Neural networks employ supervised learning following an auto-regressive modeling approach, modeling future values based on past lags or observations [6].

Numerous neural network architectures have been developed for time series forecasting, including feed-forward networks based on multilayer perceptrons (MLPs) [8], recurrent neural networks (RNNs) (e.g., long short-term memory networks (LSTMs)) [12, 43], or transformer-based

models [24]. Each of these architectures requires significant time and computational resources for hyperparameter tuning to achieve optimal performance.

1.3 Problem Definition and Objectives

Given the widespread adoption of neural networks in high-stakes applications [4, 19, 24, 43, 50], there is an increasing need for methodologies that facilitate the analysis, interpretation, and control of training behavior [11, 31]. Such approaches would support more efficient resource allocation, mitigate redundant training, and enhance the transparency of learning processes [15, 49]. Overall, there is a need for approaches that help understanding the training dynamics of neural networks as well as improving the efficiency of model development without requiring significant technical expertise.

This work is motivated by the hypothesis that performance prediction models–specifically those leveraging weight dynamics–can address these challenges. By examining the evolution of weights during training and modeling their relationship to predictive performance, this thesis aims to advance understanding of training behavior and develop practical tools for early model evaluation [32, 44].

Recent studies indicate that neural network weights encode information relevant to predictive performance [11, 32, 44]. The distribution and magnitude of these weights can serve as early indicators of model convergence and effectiveness [31]. Moreover, structural analysis of weight configurations across architectures may reveal patterns associated with successful training dynamics and configurations, informing improved initialization and design strategies [32].

We aim to address the challenge of predicting neural network performance based on model weight dynamics, focusing particularly in time series forecasting problems. Specifically, our goal is to develop a metalearning framework that can predict the final forecasting performance of a neural network based on the statistical properties and evolution of its weights [11, 32, 44]. Overall, this task can be useful for two main reasons:

1. **Understanding training dynamics:** Analyzing how weight distributions evolve during training can provide insights into the learning process of neural networks. Such patterns, if they exists, could help reduce the "black box" aspect of neural networks by establishing connections between weight states and model performance.

2. **Improving training efficiency:** If accurate performance estimates can be derived early in the training process, suboptimal model configurations and hyperparameter settings can be identified and discarded before reaching full convergence. This capability has significant practical implications for early stopping and specific operations such as algorithm selection and hyperparameter optimization. Overall, this approach can reduce the computational costs associated with model selection and hyperparameter tuning.

1.4 Research Questions

The central hypothesis of this thesis posits that metadata extracted from a neural network's weights during the training process can effectively predict its final forecasting performance. Accordingly, we formulate the following research questions to guide our work:

- **RQ1:** Can final model performance be predicted using metalearning based on weight statistics extracted from trained MLP-based neural network forecasting models? This question investigates whether weight statistics contain sufficient information to estimate performance after the model is trained.
- **RQ2:** What is the earliest point in the training process at which reliable performance predictions can be made using metalearning? In other words, can meaningful estimates be obtained from early training stages (e.g., first few epochs) or if additional information from later snapshots are necessary for more reliable predictions. Understanding this aspect of the trade-off between prediction timeliness and reliability is crucial for practical operations such as early stopping and hyperparameter optimization.
- **RQ3:** Which weight-based features are most informative for performance estimation? This question investigates which statistical and distributional properties of weights (e.g., mean, variance, norm) are most predictive of final performance. This analysis may also reveal which layers have more predictive power.

By addressing these research questions, we aim to advance the understanding of neural network training dynamics in forecasting problems, which can be useful for improving the efficiency of model development.

1.5 Contributions

To address the research questions outlined above, we develop a metalearning framework that analyzes neural network weight dynamics to predict final model performance in time series forecasting tasks. To characterize neural network weights, we compute statistical measures including mean, standard deviation, and additional distributional properties across different layers [11, 32].

Our methodology involves systematically collecting and analyzing metadata from multiple configurations of MLP neural networks trained on 6 benchmark time series forecasting datasets. Specifically, we extract and analyze weight patterns during the training process to understand their relationship with forecasting accuracy. This comprehensive analysis enables us to identify key patterns and indicators that emerge as models learn to make accurate predictions.

In summary, our contributions are a metalearning framework that leverages neural network weight statistics to predict final forecasting performance, enabling early assessment of model forecasting accuracy. This includes an empirical validation of the framework's ability to make reliable predictions of model performance, and an analysis of which weight-based features and training stages are most informative for performance prediction.

1.6 Thesis Organization

The thesis is structured into five chapters, each addressing a distinct component of the research, progressing from theoretical foundations to empirical validation and final conclusions. Chapter 1 introduces the research by defining the problem, articulating the objectives, formulating the research questions, and outlining the proposed approach.

Chapter 2 reviews the relevant literature on time series forecasting, deep learning models, and meta-learning, situating the present study within existing research and identifying key challenges in model evaluation and early prediction.

Chapter 3 details the methodology, including the definition of the forecasting task, the derivation of metadata from network weights, and the architecture and training of the meta-model.

Chapter 4 presents the experimental design, describes the datasets and evaluation metrics, and provides empirical results that address the research questions, followed by a discussion of the findings and their limitations. Chapter 5 concludes the thesis by summarizing the main contributions, discussing their implications, and suggesting directions for future research.

Chapter 2

State-of-the-Art

This chapter surveys key literature on weight-based performance prediction in deep learning models for time series forecasting. Section 2.2 reviews classical statistical approaches to time series analysis. Section 2.3 presents deep learning architectures, their formulation as supervised learning problems, and common hyperparameter optimization techniques. Section 2.4 summarizes standard datasets and evaluation metrics. Section 2.5 examines meta-learning methods, with emphasis on algorithm selection, performance prediction, and neural architecture search (NAS).

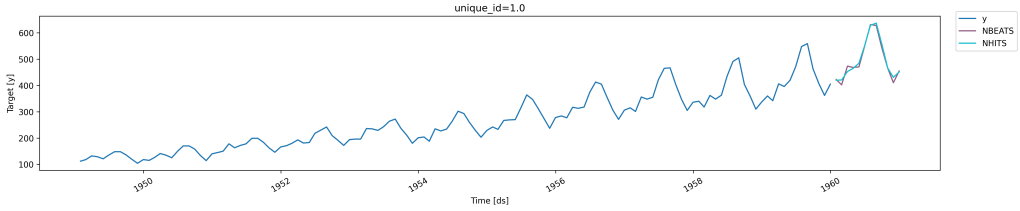
2.1 Time Series

A time series is a sequence of data points taken over time, such as stock prices, temperature readings, or sales figures. These data points are often represented graphically, with time on the horizontal axis and the variable of interest on the vertical axis. This allows analysts to identify trends, patterns, and changes over time. A line plot is often used to visualize these trends, patterns, and fluctuations, aiding in the analysis and interpretation of the data (see Figures 2.1) [36].

2.1.1 Time Series Components

Time series data can be decomposed into different components that characterize and their behavior, specifically:

Figure 2.1: Time Series Example



1. **Trend:** Trend represents the long-term movement of the data over time. It captures the overall tendency of the time series to increase, decrease, or remain stable. Trends can be deterministic, indicating a fixed pattern, or stochastic, showing more complex patterns [18].

2. **Seasonality:** Seasonality refers to periodic fluctuations or patterns that occur at regular intervals within the time series. These cycles often repeat annually, quarterly, monthly, or weekly depending on the sampling frequency and are influenced by various factors such as seasons or holidays [18].

3. **Cyclic variations:** Cyclical variations are longer-term fluctuations in the time series that do not have a fixed period. The typical example for these fluctuations are economic or business cycles [29].

4. **Remainder:** The remainder component refers to the unpredictable or random fluctuations in the data that cannot be attributed to the trend, seasonality, or cyclical variations. These fluctuations may result from random events, measurement errors, or other unforeseen factors [18].

2.1.2 Forecasting

Time series forecasting involves predicting future values of a sequence based on historical observations. Formally, given a time series $\{y_t\}_{t=1}^{T}$, where y_t denotes the observation at time t, the goal is to estimate $\hat{y}_{T+h}$ for a future time point $T+h$, using a forecasting model $f(\cdot)$:

$$\hat{y}_{T+h} = f(\{y_t\}_{t=1}^{T})$$

The objective is to approximate the underlying data-generating process in a way that minimizes forecasting error, typically evaluated using metrics such as Mean Absolute Scaled Error (MASE) or Symmetric Mean Absolute Percentage Error (sMAPE) [27, 28].

Forecasting becomes particularly challenging when applied across diverse time series datasets, where patterns, seasonality, and noise levels vary significantly. A dataset with multiple time series can thus be defined as a collection $\mathscr{D} = \{\{y_t^{(i)}\}_{t=1}^{T_i}\}_{i=1}^{N}$, where each $\{y_t^{(i)}\}$ corresponds to a distinct time series instance, possibly with varying length T_i. For forecasting tasks, state-of-the-art neural networks models have emerged due to their ability to learn from datasets composed of multiple time series. This capacity allows them to capture complex, heterogeneous temporal patterns and leverage shared structures across series. Recent studies highlight their effectiveness in improving generalization in multi-dataset settings [6], while empirical evidence from the M4 and M5 competitions [27, 28] demonstrates their superiority over traditional statistical methods in a wide range of forecasting scenarios.

2.2 Classical Time Series Models

This section provides an overview of the classical models used in time series. These models have been foundational in the field, applying varied techniques to predict future values based on historical data.

2.2.1 Naive Model

The naive model forecasts future values by repeating the most recent observation, expressed as $\hat{y}_t = y_{t-1}$, where $\hat{y}_t$ is the forecast at time t and y_{t-1} is the observed value at time $t-1$. It is simple to implement and requires minimal data preparation. While the seasonal naive variant extends this idea to account for recurring seasonal patterns, both methods assume fixed temporal repetition and tend to underperform in the presence of complex dynamics or irregular structures.

2.2.2 Autoregressive (AR) Model

The Auto-Regressive (AR) model is a cornerstone time series model that predicts future values by combining current and past values of the same time series. Specifically, it applies a linear combination of the current and past time series values along with a random error term, with the order of the model determining the number of past values used in the prediction.

Mathematically, an autoregressive model of order p, denoted as $AR(p)$, can be expressed as:

$$y_t = a + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t$$

where y_t is the value at time t, a is a constant, $\phi_1, \phi_2, \ldots, \phi_p$ are the model parameters, y_{t-1}, y_{t-2}, $\ldots, y_{t-p}$ are the lagged values, and ε_t represents white noise (random error) at time t.

2.2.3 Moving Average (MA) Model

MA models the values of time series based on a linear combination of past error terms, and the order of the moving average model, denoted by q, the order of the MA model.

As such, a model $MA(q)$ can be represented as:

$$y_t = a + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q}$$

where y_t is the value of the time series at time t, a is a constant or the mean of the time series, $\varepsilon_t, \varepsilon_{t-1}, \varepsilon_{t-2}, \ldots, \varepsilon_{t-q}$ are the white noise terms associated with the time series at times $t, t-1, t-2, \ldots, t-q$, and $\theta_1, \theta_2, \ldots, \theta_q$ are the model coefficients.

Building upon the AR and MA models described above, ARIMA (Autoregressive Integrated Moving Average) merges these approaches into a unified framework for time series forecasting. The model is denoted as $ARIMA(p,d,q)$, where p represents the number of autoregressive lag observations, d indicates the degree of differencing required for stationarity, and q specifies the order of the MA model. Due to its versatility in capturing various temporal patterns, ARIMA has

become a cornerstone model in diverse domains including retail or industry. In fact, empirical studies have shown ARIMA among the strongest univariate methods alongside Theta models, particularly in environmental and related forecasting contexts [10].

2.2.4 Exponential Smoothing

Exponential smoothing is a time series forecasting method for univariate data, which can be extended to support data with trend or seasonal components. It assigns exponentially decreasing weights for newest to oldest observations, with older data receiving less weight and newer data receiving more. Exponential smoothing has different variants that makes this approach versatile for various scenarios. For a more comprehensive read on exponential smoothing we refer the reader to the work of Hyndman et al. [16].

2.3 Forecasting with Deep Learning

Deep learning has emerged as a leading approach for time series forecasting, offering powerful capabilities for modeling time series data. This has been demonstrated in several studies [6] and benchmark competitions such as M4 [28] or M5 [27]. By leveraging datasets with multiple time series, often from various domains and with different characteristics, deep learning methods can capture patterns and long-term dependencies that traditional statistical approaches often miss. This section explores three key aspects of deep learning forecasting: how these problems are structured and formulated, the main neural network architectures used in practice, and typical strategies for tuning model hyperparameters such as learning rates and number of layers.

2.3.1 Modeling

Forecasting with deep learning is commonly framed as a supervised learning task based on auto-regression. Given a historical sequence of observations $\{y_t\}_{t=1}^{T}$, the model is trained to predict future values $\{y_{T+1}, \ldots, y_{T+h}\}$ over a specified horizon h by using a sliding window approach, where past values are used as input features to predict subsequent target values. This auto-regressive modeling technique allows the network to learn temporal dependencies by treating each time step as a training example where previous observations act as predictors for future observations [23].

Deep learning approaches typically operate by learning with time series dataset collections ($\mathscr{D}$) simultaneously, leveraging patterns and information shared across series to improve generalization. This differs from classical approaches that often treat each series independently (e.g., ARIMA [16]).

In multivariate settings, input data may also include covariates or external regressors. These can be exogenous time-dependent features (e.g., weather, holidays), categorical indicators, or positional encodings to support seasonality modeling [33]. In this work, we focus on datasets composed of multiple univariate time series, where each series represents an independent sequence of observations over time.

2.3.2 Architectures

Several neural network architectures have been developed for forecasting tasks. Recurrent Neural Networks (RNNs) [9, 43], especially Long Short-Term Memory (LSTM) [12] and Gated Recurrent Units (GRUs), were early choices for sequence modeling due to their ability to maintain temporal dependencies across time steps. More recently, Transformer-based models have gained traction in time series forecasting because of their superior performance in capturing long-range dependencies through self-attention mechanisms [24].

Despite the growing popularity of transformer-based and other architectures, methods based on Multilayer Perceptrons (MLPs) continue to show their competitiveness for time series forecasting. Architectures such as N-BEATS [37], NHITS [8], or DLinear [51]. Compared to transformers, MLPs are more efficient to train and require fewer resources, which is an important consideration in various application domains.

Hybrid approaches and ensemble methods have also proven effective. Individualized forecasting frameworks based on transfer learning adapt pre-trained models to specific time series contexts [19], while meta-learning techniques such as STG-Meta exploit spatial-temporal relationships in graph-structured time series [20]. Separately, ensemble models, particularly in domains like epidemic forecasting, enhance robustness and accuracy by combining the outputs of multiple forecasting architectures [47].

2.3.3 Model Selection

2.3.3.1 Hyperparameter Tuning

Neural networks usually require dataset-specific configuration of hyperparameters such as learning rates, batch sizes, number of layers, and hidden units. Adequate configurations critically influence convergence behavior and forecasting accuracy. Manual tuning is often infeasible in practice due to the high dimensionality of the search space.

Consequently, automated hyperparameter optimization techniques have become standard. Common approaches include random search (e.g. [21]) or Bayesian optimization, where the latter balances exploration of the parameter space with exploitation of promising regions [52]. These methods are particularly useful when training costs are high and model evaluation is time-consuming. Another widely used strategy in hyperparameter tuning is early stopping, which denotes the process of halting training when the model's performance on a validation set stops improving. This approach prevents overfitting and reduces unnecessary computation by stopping training before the model begins to memorize noise.

2.3.3.2 Neural Architecture Search

Neural Architecture Search (NAS) is a subfield of meta-learning focused on automating the design of neural network structures. Instead of manually selecting architectural components such as the

number of layers, neurons, or activation functions, NAS algorithms explore these configurations to optimize model performance.

The Bayesian NAS framework introduced by Valkanas et al. [45] presents a principled approach for discovering optimal graph neural network (GNN) architectures, jointly modeling architecture, parameters, and structure within a unified probabilistic framework. Their method explicitly accounts for uncertainty in design decisions, outperforming deterministic baselines across standard datasets.

In parallel, Zimmer et al. [52] developed Auto-PyTorch, an AutoDL framework for tabular data that automates the joint optimization of network architectures and training hyperparameters. While not specifically tailored for time series, the framework illustrates the broader applicability of meta-learning principles in NAS.

NAS has been employed in time series applications to automate the neural network architectures that are suited to data with a temporal structure. Rakhshani et al. [40] applied NAS to time series classification by evolving deep residual networks, demonstrating performance improvements over conventional baselines. More recently, Lyu et al. [25] introduced ONE-NAS, an online framework that adapts recurrent neural network architectures in real time to accommodate nonstationary and multivariate time series.

2.4 Evaluation

The evaluation of forecasting models is a fundamental component of empirical research in time series analysis. It facilitates the objective comparison of competing methodologies and informs methodological refinement. Due to the temporal dependencies inherent in time series data, conventional evaluation techniques must be adapted to preserve causality and avoid information leakage. This section delineates standard validation strategies and the metrics commonly employed to assess predictive performance.

2.4.1 Data Partinioning

In contrast to conventional k-fold cross-validation, which assumes the independence and identical distribution of observations, time series forecasting necessitates evaluation protocols that account for temporal autocorrelation. Random partitioning of the data violates the chronological structure, potentially resulting in biased estimates of predictive accuracy [34].

A widely adopted approach is *rolling-origin* or *walk-forward validation*, wherein the training set is progressively expanded and the validation set is advanced in a sequential manner. This method simulates the operational constraints of real-world forecasting, where only historical data are available at prediction time. By reevaluating the model at each step and evaluating on the subsequent time point or forecasting horizon, this strategy maintains the integrity of temporal ordering and provides a robust estimate of out-of-sample performance [7]. When using datasets composed of multiple time series, the typical approach is to leave the last h observations of each time series for testing, where h is the forecasting horizon [42].

2.4.2 Evaluation Metrics

2.4.2.1 Classification and Regression

When it comes to evaluating a neural network's performance, the type of metrics we use depends on the task performed. For classification tasks, we use several key metrics to tell us how well the model is doing [13]. The ROC AUC score, for instance, measures how effectively the model can distinguish between different classes. A higher score means it's better at telling them apart. Another metric, Log Loss, assesses the accuracy of the predicted probabilities, applying a heavy penalty if the model is very confident about a wrong answer. Then there is the F1 Score, which calculates the harmonic mean of precision and recall, making it particularly useful when we're dealing with imbalanced datasets. Finally, Accuracy provides a straightforward percentage of correct predictions, though it can be misleading if the classes are not evenly distributed.

For regression tasks, where the goal is to predict a specific value, our performance metrics generally look at either error or correlation. Error-based metrics, as covered in section 2.4.2.2, are all about measuring the size of the prediction mistakes. A common example is the Root Mean Square Error (RMSE), a statistical metrics that tells us the average difference between the model's predictions and the actual values. We also use the Coefficient of Determination, represented as R^2, which indicates the proportion of the variance in the actual outcomes that our model is able to explain. On the other hand, correlation-based metrics evaluate the strength of the relationship between the predicted and actual values. The Pearson correlation coefficient is used to measure a direct linear relationship, but it works best with data that is continuous and roughly normalized [1]. If the relationship isn't linear, or if the ranking of the values is more critical than their precise amounts, non-parametric alternatives like Kendall's τ and Spearman's rank correlation are more robust choices, as they assess monotonic relationships using data ranks instead of the raw data itself [1].

2.4.2.2 Forecasting

Evaluation is a crucial aspect of time series forecasting, enabling practitioners to assess the performance of different models. It involves comparing predicted values with actual observed values over a given period. Common evaluation metrics for forecasting include Symmetric Mean Absolute Percentage Error (sMAPE) and Mean Absolute Scaled Error (MASE):

$$\text{sMAPE} = \frac{100\%}{n} \sum_{i=1}^{n} \frac{|\hat{y}_i - y_i|}{(\hat{y}_i + y_i)/2}$$

$$\text{MASE} = \frac{1}{n} \sum_{i=1}^{n} \frac{|y_i - \hat{y}_i|}{\frac{1}{n-m} \sum_{i=m+1}^{n} |y_i - y_{i-m}|}$$

where $\hat{y}_i$ and y_i are the forecast and actual value for the i-th instance, respectively, n is the number of observations, and m is the seasonal period. These and other metrics are usually computed across all available prediction points, which include multiple time steps, forecasting horizons, and time

series. As outlined by Hyndman et al. [17] and others [27, 28], these metrics are fundamental in practical forecasting challenges.

2.5 Metalearning

Metalearning [5] is a subfield of machine learning that applies machine learning techniques to improve the development and performance of machine learning models themselves by analyzing patterns across different learning tasks and their solutions, meta-learning systems can automatically select, configure, and optimize models for new problems, presenting a systematic approach to automating machine learning pipelines.

2.5.1 Algorithm Selection Problem

In metalearning, *algorithm selection* is a foundational problem, which involves identifying the most suitable algorithm for a new task based on past performance over similar problems by extracting features from datasets. This features can range from statistical properties, seasonality indicators, or complexity measures, allowing meta-models to learn how to recommend models with high expected accuracy [46]. This reduces reliance on exhaustive model comparisons and supports more efficient experimentation pipelines.

This problem is highly relevant for time series forecasting, where the diversity of series patterns and frequencies necessitates tailored model configurations [38]. Metalearning helps address this variability by capturing cross-task regularities and facilitating informed model initialization.

2.5.2 Predicting Performance

Performance prediction is a key component of the algorithm selection problem, where the goal is to predict which algorithm will perform best for a given task by analyzing the characteristics of a problem and using historical performance data across different algorithms and datasets. Then we can build a model and estimate how well a particular configuration or algorithm is likely to perform. This process enables more informed algorithm selection decisions [5, 15].

The study by Unterthiner et al. [44] explores the concept of predicting neural network accuracy using only the weights of trained models, without requiring input data evaluation. The research demonstrates that simple weight statistics are sufficient for accurate predictions. This insight applies across both under-parameterized and over-parameterized neural network settings, highlighting the robustness of the approach. The researchers introduce a formal framework for this predictive task, supported by the release of a large dataset comprising 120,000 convolutional neural networks (CNNs) [14, 44]. Regression algorithms, including gradient boosting machines (GBM) and deep neural networks [23, 31], are employed to demonstrate the feasibility of this method.

However, the work also identifies limitations, such as the exclusion of numerically unstable models and the limited exploration of properties across various datasets and architectures. The

findings raise important questions about domain shifts and the relationship between weight properties and model performance. Similarly to Unterthiner et al. [44], our work also attempts to predict the performance of neural networks, but in the context of time series forecasting tasks.

Building upon these ideas, Martin et al. [32] proposed *WeightWatcher*, a diagnostic tool grounded in random matrix theory that quantifies the generalization ability of deep networks by analyzing the spectral properties of weight matrices. This approach supports weight-based performance estimation even in complex and overparameterized models.

Our work extends this research to the time series domain. Specifically, we investigate whether statistical descriptors of weight dynamics from partially trained MLPs can predict forecasting accuracy. This enables early stopping and model triage, allowing the rejection of poor-performing configurations before training completes.

Chapter 3

Methodology

Neural networks, despite their strong predictive performance, present considerable challenges in configuration [39]. Training is computationally intensive, highly sensitive to hyperparameter settings, and frequently lacks interpretability. Small parameter changes such as in the learning rate, weight initialization, or other architectural parameters (e.g., number of layers or hidden units) can result in substantially different forecasting accuracy. This sensitivity renders model selection and performance estimation complex tasks requiring careful methodological design and domain expertise.

In this context, the objective of this study is to examine and model the relationship between neural network weight statistics and forecasting performance. The aim is to enhance the efficiency and reliability of model selection in time series forecasting by enabling the early identification and exclusion of configurations with low predictive potential, thereby avoiding the computational cost of full training.

To address this, we propose a meta-learning framework that estimates final forecasting accuracy based on statistical features derived from model weights during training. This approach allows for performance prediction potentially without complete model convergence, by learning a mapping from internal model representations to final accuracy. The resulting meta-model serves as a surrogate for direct performance evaluation and facilitates future applications such as early stopping, architecture optimization, and automated model selection.

Our methodology is structured into two main components: (i) metadata collection (Section 3.1) and (ii) performance prediction (Section 3.2).

3.1 Metadata Collection

To enable predictive modeling via meta-learning, we construct a structured metadata dataset from multiple neural network training runs. Each record corresponds to a unique combination of hyperparameters and datasets, with features extracted at multiple training checkpoints. These features include descriptive statistics (e.g., mean, standard deviation), structural properties (e.g., norms),

and temporal patterns in weight evolution. The resulting dataset is used to train a meta-model capable of forecasting a neural network's final performance based on weight-based information.

3.1.1 Workflow

Figure 3.1: Metadata collection process.

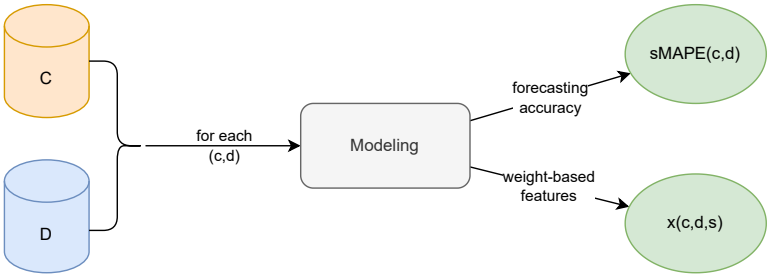


Figure 3.1 illustrates the overall workflow for metadata collection. This process involves following elements: model configurations, datasets, metadata extracted metadata vector and the evaluation metric.

Let $\mathscr{C}$ denote the set of all defined model configurations, and let $\mathscr{D}$ represent the set of time series datasets used for training and evaluation.

For each pair (c,d), where $c \in \mathscr{C}$ and $d \in \mathscr{D}$, a neural network model M_c is trained and evaluated over a fixed number of steps $\mathscr{S}$, for a time series Y_d.

$$M_c \leftarrow z(Y_d)$$

At a specified training step $s \leq \mathscr{S}$, metadata is extracted in the form of weight-based information vector, denoted by $x(c,d,s)$. Each vector $x(c,d,s)$ comprises descriptive statistics of the model's weights, hyperparameter encoding, and dataset characteristics, all extracted independently of the training pipeline for use in predictive modeling. This weight-based information, which serves as the explanatory variables for metalearning, is collected at a specified training step s. This flexibility enables the evaluation of the approach across different training stages. The target variable for metalearning is the model's final forecasting error $e(c,d)$, computed upon completion of training, while the explanatory variables can be obtained at arbitrary points during the training process.

As such, the metadata dataset $\mathscr{D}_{\text{meta}}$, that aggregates the descriptive statistics of the model's weights can be described as:

$$\mathscr{D}_{\text{meta}} = \left\{ (x(c,d,s), e(c,d)) \;\middle|\; c \in \mathscr{C},\ d \in \mathscr{D},\ s \leq \mathscr{S} \right\},$$

where:

- $x(c,d,s) \in \mathbb{R}^p$ is the explanatory feature vector, comprising weight-based summary statistics, hyperparameter encoding and dataset descriptors;

- $e(c,d) \in \mathbb{R}$ is the target metric, i.e. the final forecasting error of M_c on Y_d after $\mathscr{S}$ steps;

3.1.2 Weight-based Features

At a given training step s, we extract features based on the weights of the neural network applied to configuration c on dataset d. These features characterize the statistical and structural properties of the model parameters at that training step s, providing a snapshot of the training dynamics. The metrics are computed directly from the 2D tensor values of each weight matrix. Given that the architecture remains constant across experiments, comprising l hidden layers trained over $\mathscr{S}$ steps, a separate set of metrics is calculated for each of the corresponding weight tensors. An overview of the extracted weight features is presented in Table 3.1.

Table 3.1: Weight Statistics (per layer)

Category	**Description**
Summary stats	Mean, standard deviation, min, max
Norms	Frobenius norm, spectral norm
Distribution	Power-law alpha, weighted alpha
Model Variance	Variance of the weight matrix

The use of norm and distribution-based features is motivated by the WeightWatcher framework [32], an open-source diagnostic tool grounded in Random Matrix Theory and the theory of Heavy-Tailed Self-Regularization. These metrics, such as spectral norm, Frobenius norm, and the power-law exponent alpha, have been empirically linked to generalization performance in deep learning models. They enable model quality assessment using only information from the weight matrix, making them well-suited for inclusion in a metamodel seeking to predict performance from weight dynamics alone.

3.1.3 Performance Estimation

Besides the weight-based features, computed at an arbitrary training step s, we also perform a performance estimation step to quantify the forecasting accuracy of each trained model, given configuration c and dataset d. This step provides the target variables for the meta-learning framework.

Two types of evaluation metrics are used: a continuous regression metric and a binary classification indicator. The regression metric is the Symmetric Mean Absolute Percentage Error (sMAPE), a widely adopted measure of forecasting accuracy that accounts for both scale and relative error [28]. The classification metric is a Boolean indicator that denotes whether the model's forecasting performance (sMAPE) exceeds that of a Seasonal Naive baseline. An overview of these evaluation metrics is presented in Table 3.2.

Table 3.2: Evaluation Metrics

Category	**Type**	**Description**
Regression	Continuous	sMAPE
Classification	Boolean	Indicates whether the model's forecasting performance is better than the Seasonal Naive baseline

In addition to the evaluation metrics, we also record contextual information related to each experiment, including the hyperparameters used during training and dataset properties. The complete set of hyperparameters and dataset attributes is summarized in Table 3.3.

Table 3.3: Hyperparameters and Dataset Properties

Category	**Feature**	**Description**
Hyperparameters	Hidden size	Number of neurons per layer
	Learning rate	Learning rate used in training
	Batch size	Size of each mini-batch
	Scaler type	Preprocessing strategy
	Random seed	Seed for weight initialization
Dataset Properties	Frequency	Data frequency (e.g., monthly, quarterly)
	Seasonality	Seasonality strength indicator

These weight-based statistics, evaluation metrics and contextual features together constitute the explanatory and target variables stored in the metadata dataset $\mathscr{D}_{\text{meta}}$.

3.2 Metalearning Performance

The central problem addressed in this research is the prediction of neural network performance in time series forecasting tasks. Specifically, we aim to estimate a model's final forecasting accuracy using training information, captured through the statistics of its weight configurations collected during the training process.

As outlined previously, our approach is structured into two main components: (i) metadata collection and (ii) performance prediction. The first component involves constructing a dataset comprising weight-based features, hyperparameters, and dataset characteristics extracted from multiple neural network training runs. The second component focuses on training a meta-learning model to predict forecasting performance based on these features.

Once the metadata dataset $\mathscr{D}_{\text{meta}}$ is constructed, the next step is to train a meta-learning model capable of predicting the final forecasting performance of a neural network. This is framed as a supervised learning task, using either a regression or classification objective depending on the chosen target as described before.

3.2.1 Meta-model Formalization

Formally, this becomes a supervised learning task where the input is a vector of weight-derived features, and the output is the final forecasting error of the model.

Let $x(c,d,s)^{(i)} \in \mathbb{R}^p$ denote the feature vector extracted from training stages of the i-th model, and let $e(c,d)^{(i)} \in \mathbb{R}$ represent its corresponding final forecasting error. The meta-learner then learns the function:

$$g(x(c,d,s)) \approx e(c,d)$$

where:

- $g : \mathbb{R}^d \to \mathbb{R}$ for regression (e.g., final sMAPE),
- or $g : \mathbb{R}^d \to \{0,1\}$ for classification (e.g., *better-than-baseline*).

This function is trained to minimize the prediction loss $\mathscr{L}(g(\mathbf{x}^{(i)}), e(c,d)^{(i)})$ across a set of training examples.

3.2.2 Training Procedure

To evaluate the predictive utility of weight-based features for forecasting accuracy, we train a meta-learning model using the structured metadata dataset constructed from neural network training runs. Each instance in the dataset corresponds to a metadata vector $x(c,d,s)$, where c denotes the model configuration, d the dataset, and $s \leq \mathscr{S}$ a training step. The target is either the final forecasting error (sMAPE) or a binary indicator of whether the model outperforms a Seasonal Naive baseline.

To simulate real-time model selection scenarios, we adopt a stage-wise training strategy. A sequence of training stages $\mathscr{S} = \{s_1, s_2, \ldots, s_n\}$ defines checkpoints from which features are incrementally included. At each stage s_i, only features corresponding at steps $\{s_1, \ldots, s_i\}$ are used to train a new meta-model, enabling an assessment of how early in training accurate performance predictions can be made.

Chapter 4

Experiments

4.1 Research Questions

The experimental phase of this work is structured around three core research questions. These questions are designed to evaluate the effectiveness and efficiency of using meta-learning to predict the final forecasting performance of neural networks from metadata extracted from the weights of neural networks. Each question targets a specific dimension of the proposed methodology: **RQ1** investigates if statistics from a neural network's weights can predict its final forecasting performance; **RQ2** explores how early in training reliable performance predictions can be made, by assessing when weight information stabilizes adequately for forecasting; **RQ3** analyzes which weight-based features possess higher predictive value.

4.1.1 Datasets

We employ six datasets from three forecasting benchmarks: GluotonS M1 [30], M3 [26], and Tourism [2]. These datasets are standard in the time series forecasting literature and span multiple domains, including economics, industry, demography, and tourism. A detailed summary of their properties is provided in Table 4.1. We restrict our analysis to low-frequency univariate time series, specifically monthly and quarterly series.

Across all datasets, we use a forecasting horizon of $h = 12$ for monthly series and $h = 8$ for quarterly series. The input window (lag length) is set to 24 and 8 for monthly and quarterly series, respectively, which is equivalent of two full seasonal periods.

Overall, the combined dataset comprises 3,797 time series and a total of 409,602 observations supporting a robust and diverse evaluation of our meta-learning framework across a wide range of series lengths, domains, and seasonal structures.

Table 4.1: Summary of the datasets: number of series, total observations, seasonal period, and forecasting horizon.

Dataset	**# Time Series**	**# Observations**	**Seasonal Period**	**Horizon (h)**
Gluonts M1 Monthly	617	44,892	12	12
Gluonts M1 Quarterly	203	8,320	4	8
M3 Monthly	1,428	167,562	12	12
M3 Quarterly	756	37,004	4	8
Tourism Monthly	366	109,280	12	12
Tourism Quarterly	427	42,544	4	8
Total	3,797	409,602	-	-

For performance estimation of forecasting models for a given configuration, we perform a holdout strategy aligned with the defined horizons. For each time series, we reserve the last h observations as the test set, $h = 12$ for monthly series and $h = 8$ for quarterly series. The remaining observations form the training set ensuring consistent evaluation across datasets and realistic forecasting conditions.

4.1.2 Metadata Extraction

To ensure reproducibility and systematic evaluation, each training and testing run collected relevant metadata, including hyperparameters, preprocessing configurations, training time, validation metrics, and random seed values.

Table 4.2 presents the defined hyperparameter space, encompassing architectural and optimization parameters:

- *Hidden Size*, number of units in each hidden layer;
- *Learning Rate*, step size for gradient updates during training;
- *Batch Size*, number of samples per training update;
- *Scaler Type*, method used to normalize input data;
- *Random Seed*: initialization seed for reproducibility to ensures consistent results across runs.

The MLP architecture has fixed hidden layers, $l = 3$, and training steps, $S = 500$, to ensure consistent training and manageable computational demands. This range was selected to enable a broad empirical assessment of model performance.

Metadata was stored alongside model checkpoints. We considered a total of $3 \times 2 = 6$ dataset configurations and $4 \times 1 \times 1 \times 3 \times 3 \times 4 \times 5720$ hyper-parameter configurations, yielding $6 \times 720 = 4320$ total combinations. Metrics were collected at defined training steps (start, 10 steps, 25 steps, 50 steps, 100 steps, 200 steps, 300 steps, 400 steps, and 500 steps) and a custom callback was

Table 4.2: Hyperparameter Space

Hyperparameter	**Values**
Hidden Size	8, 16, 32, 64
Maximum Steps	500
Number of Layers	3
Learning Rate	10^{-3}, 5×10^{-4}, 10^{-4}
Batch Size	16, 32, 64
Scaler Type	Identity, Standard, Robust, Min-Max
Random Seed	42, 123, 456, 789, 1011

implemented to log model weights, optimizer states, and evaluation metrics at these intervals, such as statistical and spectral properties of weight matrices. These metrics are summarized in Table 4.3. The information collected supports model interpretability and comparative analysis across training configurations.

Table 4.3: Metrics Extracted During Training by Custom Callback

Metric	**Description**
Mean / Median / Variance / Std Dev	Basic statistics computed for each weight matrix
Min / Max	Minimum and maximum weight values
Frobenius Norm	Measures the overall magnitude of a weight matrix by taking the square root of the sum of the squares of all weight entries. Reflecting the scale of the matrix.
Spectral Norm	Largest singular value of the weight matrix. Indicates how much the matrix can stretch a vector in any direction
Power-law alpha α	Exponent from power-law fit to the eigenvalue distribution
Weighted alpha $\hat{\alpha}$	Adjusted exponent scaled by mean-to-median ratio
Model Variance	Variance across all model parameters

Tables 4.4 and 4.5 summarize the distribution of these scores across datasets.

Table 4.4: Proportion *is_better* Score by Dataset

Dataset	**Group**	**% Better**
Gluonts	Monthly	70.14
	Quarterly	93.06
M3	Monthly	82.64
	Quarterly	59.03
Tourism	Monthly	9.72
	Quarterly	18.06
Overall		55.44

The proportion of instances for which models correctly predict superior performance (*is_better*, a boolean indicator of whether the model outperforms the Seasonal Naive baseline) varies greatly by dataset and seasonality (Table 4.4). Gluonts exhibits the highest success rates (93.06% quarterly, 70.14% monthly), followed by M3 (59.03% quarterly, 82.64% monthly), while Tourism yields substantially lower rates (18.06% quarterly, 9.72% monthly). The overall success rate is

Table 4.5: Mean sMAPE Score by Dataset

Dataset	Group	Mean sMAPE
Gluonts	Monthly	0.1038
	Quarterly	0.0877
M3	Monthly	0.0802
	Quarterly	0.0698
Tourism	Monthly	0.1385
	Quarterly	0.1406
Overall		0.1034

55.44%. Mean sMAPE scores reflect a similar ordering (Table 4.5), with M3 quarterly attaining the lowest error (0.0698) and Tourism quarterly the highest (0.1406). Gluonts achieves moderate errors (0.0877 quarterly, 0.1038 monthly), producing an overall mean sMAPE of 0.1034. These results suggest that the models are most effective on datasets with relatively stable seasonal patterns and lower forecast difficulty, and less effective on highly volatile series such as those in the Tourism dataset.

4.1.3 Metamodel Training

The metamodel was trained to predict model performance based on the evolution of weight matrix statistics collected at various training steps. We utilize a stage-wise training strategy with checkpoints $\mathscr{S} = \{s_1, s_2, \ldots, s_n\}$, incrementally including features to evaluate performance prediction. Two separate prediction tasks were considered: a regression task, using Symmetric Mean Absolute Percentage Error (sMAPE) as the target, and a classification task, predicting whether a model outperformed a Seasonal Naive baseline (*is_better*).

The input features were derived from statistical and spectral metrics per weight matrix, excluding identifiers and target variables. Shape-based features and dataset identifiers were dropped to prevent leakage. Categorical features, such as scaler type, were encoded numerically.

Models were trained using the Random Forest algorithm, with the implementation available in the *xgboost* Python package. Each training configuration was evaluated using Group K-Fold cross-validation, stratified by dataset group, to ensure reliable performance estimates.

This stagewise analysis allowed us to assess the predictive capacity of early weight statistics and identify which features and timepoints were most informative for metamodel performance.

4.2 Results

This section presents the empirical results for each of the research questions. Performance is evaluated at multiple training checkpoints, allowing us to track how predictive power evolves over time. We report metrics for both classification and regression tasks, using metadata extracted from MLP models trained on diverse time series datasets.

4.2.1 Can we predict a neural network's final forecasting performance using metadata derived from its weight statistics?

To address RQ1, we evaluate the meta-model using metadata collected at the end of the training stage, $S = 500$, where the signal is presumed strongest. Tables 4.6 and 4.7 summarize the results. Classification metrics are reported globally because the task involves learning a general decision boundary for whether a model outperforms the baseline, regardless of dataset. Stratifying by dataset would reduce sample size per group and obscure overall model discriminative power, which is best evaluated across the full distribution of cases.

Table 4.6: Final stage evaluation metrics for classification tasks.

Metric	Mean	Std
Accuracy	0.5463	0.1691
ROC AUC	0.6778	0.1740
Log Loss	0.9144	0.3075
F1 Score	0.5253	0.2189

Table 4.7: Final stage evaluation metrics for regression tasks, by dataset.

Dataset	Season	MAE	MSE	R^2	Pearson	Kendall	Spearman
Tourism	Quarterly	0.0724	0.0158	-0.1797	0.3649	0.6722	0.8541
	Monthly	0.0555	0.0055	-1.0078	0.1817	0.4642	0.6285
M3	Quarterly	0.0799	0.0152	-1.7976	0.5736	0.5972	0.7883
	Monthly	0.0604	0.0052	-2.9230	0.5422	0.7542	0.8866
Gluonts	Quarterly	0.1024	0.0251	-0.6802	0.2571	0.5425	0.7380
	Monthly	0.0431	0.0099	0.1135	0.3422	0.5107	0.6845

The classification results indicate that the meta-model can moderately distinguish between models that surpass the seasonal naive baseline and those that do not, with a ROC AUC of 0.68. Accuracy and F1 Score have a high standard deviation, suggesting that while the task is challenging, weight statistics contain useful signals.

The regression performance varies by dataset and seasonality. For quarterly series, mean absolute error (MAE) ranges from 0.0724 (Tourism) to 0.1024 (Gluonts), and mean squared error (MSE) from 0.0152 (M3) to 0.0251 (Gluonts). For monthly series, MAE lies between 0.0431 and 0.0604, and MSE between 0.0052 and 0.0099. Rank-based correlations are uniformly strong: Spearman's ρ ranges from 0.6285 (monthly Tourism) to 0.8866 (monthly M3), and Kendall's τ ranges from 0.4642 (monthly Tourism) to 0.7542 (monthly M3). Only the monthly Gluonts configuration attains a positive R^2 (0.1135); all other configurations exhibit negative R^2 (minimum -2.9230 for monthly M3), indicating limited accuracy in absolute-error estimation. Despite this, the combination of low MAE/MSE and high rank correlations demonstrates that weight-statistic metadata can effectively screen and rank forecasting models.

4.2.2 How early in the training process can reliable performance predictions be made?

To address RQ2, we evaluate the evolution of prediction quality over user-defined training checkpoints, $S = \{0, 10, 25, 50, 100, 200, 300, 400, 500\}$. For each metamodel experiment, features from an increasing number of training stages were cumulatively included, enabling analysis of how early training dynamics correlate with final performance. Performance is aggregated by dataset and seasonality in order to assess how predictive accuracy varies across different forecasting domains.For this reason the metrics are computed separately for each dataset group and then averaged to capture general trends. This allows us to determine at which point the metadata becomes sufficiently informative to produce stable and reliable performance estimates. Figures 4.1–4.5 show how classification and regression metrics evolve during training.

Figure 4.1: Evolution of the ROC AUC score over training steps.

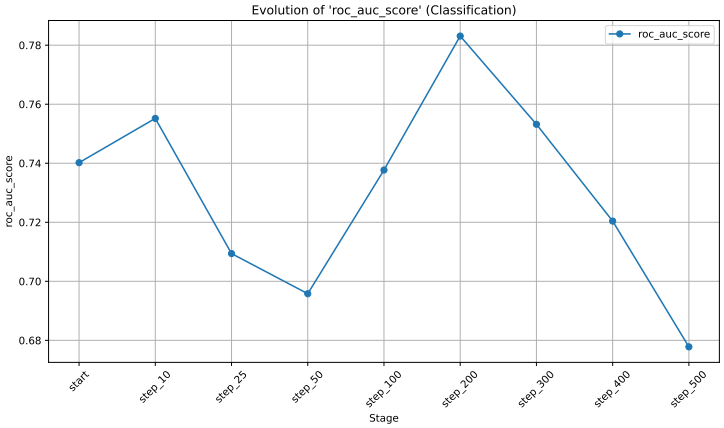
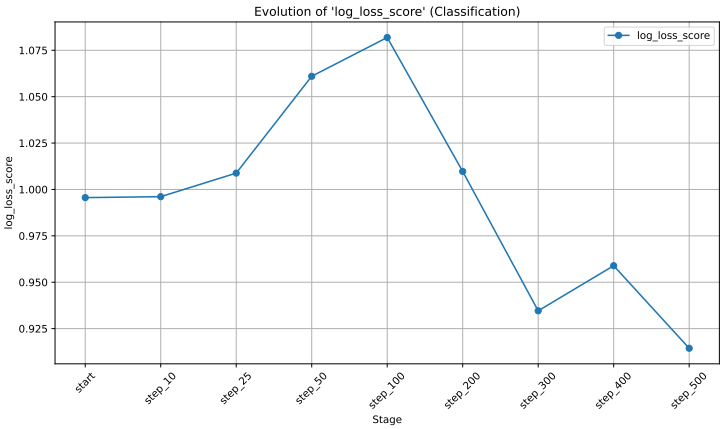
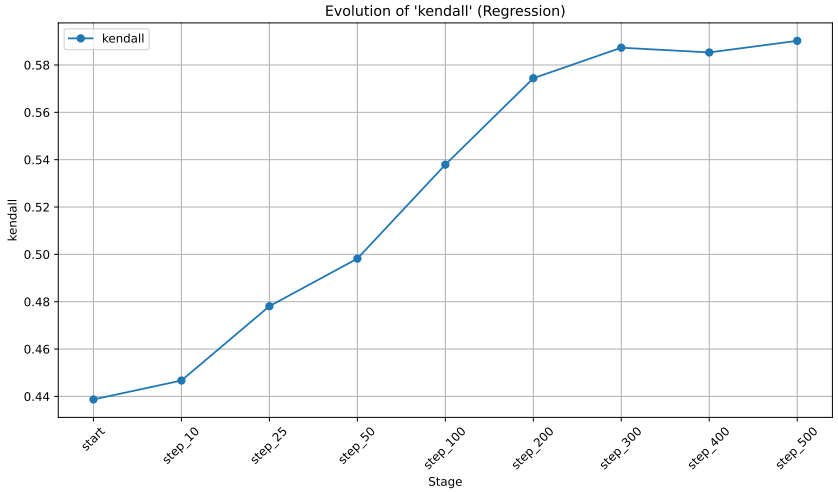


Figure 4.2: Evolution of the LOG LOSS over training steps.



The ROC AUC score (Figure 4.1) improves from the initial stage (0.7402 at *start*) and peaks early at step 10 (0.7552), before declining at steps 25 and 50 (down to 0.6958). It briefly recovers at step 100 (0.7377) and then reaches its maximum at step 200 (0.7835), before consistently

Figure 4.3: Evolution of the Kendall correlation over training steps.



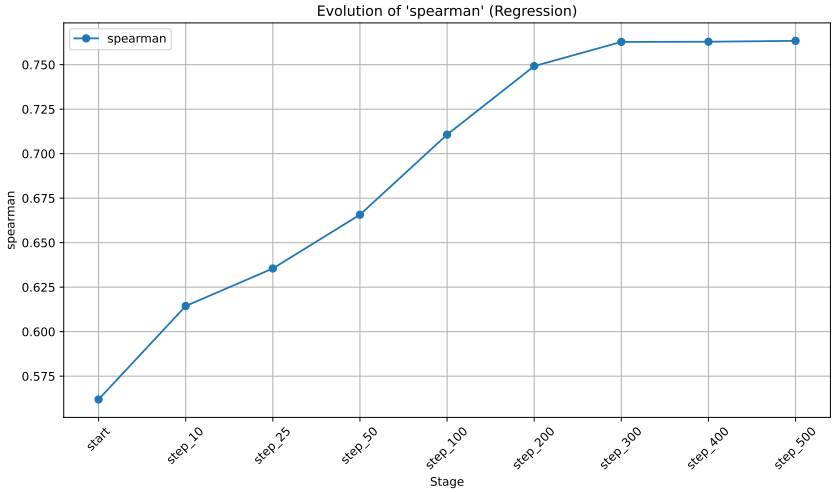
falling through steps 300 to 500 (bottoming at 0.6787). This trend suggests that predictive signal becomes most discriminative around step 200, beyond which the added training signal does not translate to better classification performance and may even lead to reduced generalization. However, the results are inconsistent, changing direction as the number of steps increases, which indicates unstable performance across training progression.

The log loss score (Figure 4.2) reveals a different pattern. It remains relatively flat between *start* and step 25 (around 0.995–1.009), then increases notably at steps 50 (1.0610) and 100 (1.0819), indicating less confident probabilistic predictions. However, after step 100, log loss improves significantly, dropping to 0.9309 at step 300, slightly rising at step 400 (0.9526), and reaching its lowest point at step 500 (0.9144). This decline shows that model calibration benefits substantially from longer training, and that later-stage metadata helps produce more accurate probability estimates.

For regression tasks, both Kendall and Spearman rank correlation metrics (Figures 4.3 and 4.4) exhibit consistent improvement throughout the training process. Kendall correlation starts at 0.4387 and steadily rises to 0.5902 by step 500, with notable gains after step 100. Spearman correlation follows a similar trajectory, increasing from 0.5619 at *start* to 0.7634 at step 500, showing meaningful jumps between step 50 and step 200. This indicates that the meta-model becomes progressively better at ranking forecasting models as training progresses.

In contrast, Pearson correlation (Figure 4.5) is more volatile. It peaks early at step 25 (0.4371), drops sharply at step 100 (0.2853), and fluctuates thereafter, ending at 0.3770. This instability highlights the sensitivity of linear correlation to value scaling and outliers, which are not as problematic for rank-based metrics like Kendall and Spearman.

Figure 4.4: Evolution of the Spearman correlation over training steps.



4.2.3 Which metadata features are most important for accurate performance prediction?

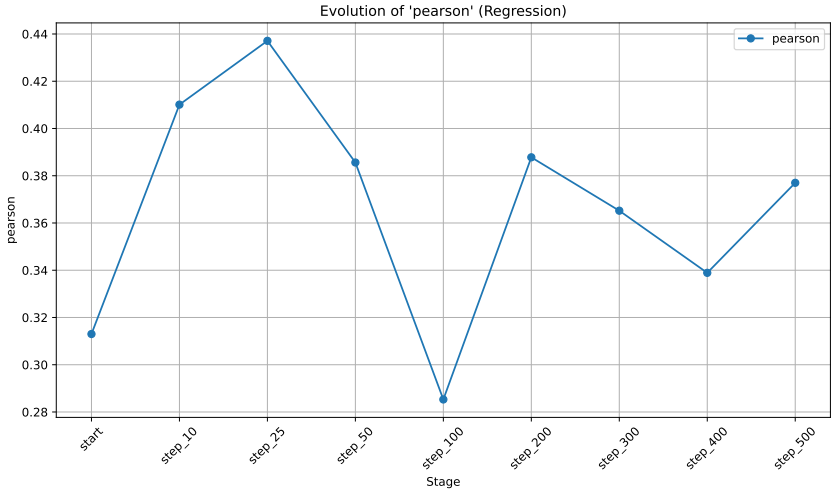
To answer RQ3, we analyzed feature importance derived from meta-models trained using the full training data at step 500. Separate rankings were generated for the classification and regression tasks to reflect their differing performance objectives and signal structures.

Figure 4.6 displays the top individual features contributing to classification performance. The most influential features are predominantly drawn from later training stages and are concentrated in the output layer. In particular, the weighted power-law exponent of the output layer consistently appears among the top-ranked features, indicating that measures of spectral complexity and weight distribution shape in the final layer are highly informative for binary prediction. This highlights the importance of output-layer structural regularity for identifying models that exceed the baseline threshold.

Figure 4.7 shows the corresponding feature importance rankings for the regression task. Here, the dominant features originate from earlier and intermediate layers, most notably the Frobenius norm of the first hidden layer and the unweighted power-law exponent of the third hidden layer. These features appear consistently across multiple training checkpoints, suggesting that magnitude-based and spectral descriptors in lower layers serve as strong early indicators of forecasting error.

Grouped importance values, illustrated in Figures 4.8 and 4.9, confirm these observations. For classification, the weighted power-law exponent group from the output layer contributes the largest cumulative signal, particularly from step 300 onward. For regression, the most important group is the Frobenius norm of the first hidden layer, which maintains a high relative importance across early to mid training steps. This suggests that early-stage weight magnitudes in lower layers encode meaningful information for forecasting accuracy estimation, even before training convergence.

Figure 4.5: Evolution of the Pearson correlation over training steps.



4.3 Discussion

The experimental results substantiate the viability of the proposed meta-learning framework for forecasting performance prediction using weight-derived metadata from MLP models. Each research question isolates a distinct aspect of the framework—predictive capacity, temporal efficiency, and feature relevance—and collectively, they affirm its practical applicability.

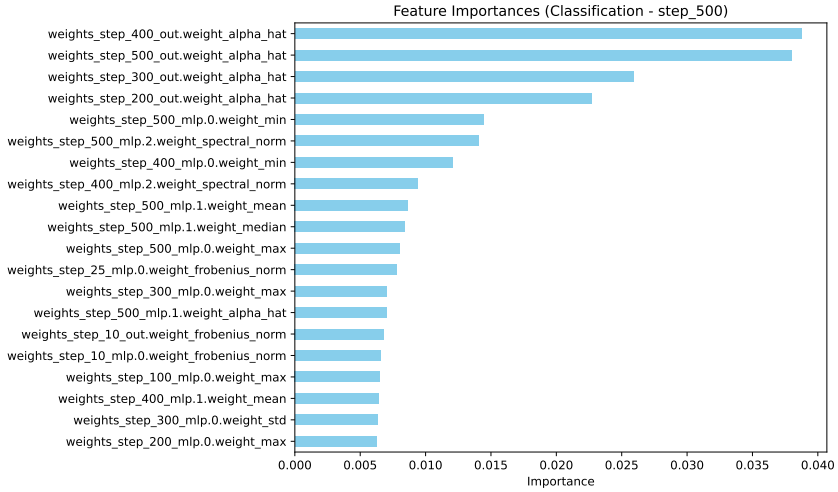
For **RQ1**, the findings confirm that weight-based metadata encodes sufficient information to predict final forecasting performance. This supports the core hypothesis and demonstrates that the meta-model can approximate model performance without full training, thereby enabling applications such as model ranking, selection, and early termination. Although prediction accuracy shows room for improvement, the results validate the framework's overall effectiveness.

In the context of **RQ2**, performance metrics computed across training stages indicate that reliable predictions emerge well before training concludes. Classification accuracy, as measured by ROC AUC, reaches its maximum around step 200, although with variations that show inconsistent results, while log loss continues to improve through step 500, reflecting better probabilistic calibration over time. Likewise, rank-based correlation metrics (Spearman and Kendall) stabilize around step 200, indicating that model ranking becomes reliable at this point. These trends highlight a trade-off between early efficiency and late-stage precision: early checkpoints suffice for coarse model filtering, whereas later stages provide refined predictive estimates.

For **RQ3**, the feature importance analysis reveals task-specific patterns in the predictive value of weight-based metadata. In the classification task, the most informative features are derived from the output layer at later training stages, particularly the weighted power-law exponent. These features reflect spectral complexity and scaling behavior, and their prominence suggests that structural characteristics of the final-layer weights are essential for distinguishing whether a model outperforms a naive baseline.

By contrast, regression prediction relies more heavily on early-layer signals. Features such

Figure 4.6: Final ranking of feature importance for the classification task.



as the Frobenius norm of the first hidden layer and the unweighted power-law exponent of intermediate layers consistently rank highly. These metrics describe global weight magnitude and spectral shape, and their stability across training stages indicates that they capture early indicators of generalization capacity and convergence behavior.

These findings underscore the importance of aligning feature selection with task objectives. Classification performance estimation benefits from late-stage complexity measures in the output layer, while regression accuracy is more strongly associated with early-layer norms and distributional regularities in lower layers. This distinction has practical implications for designing efficient meta-learning pipelines that prioritize the most relevant statistical descriptors based on the prediction target.

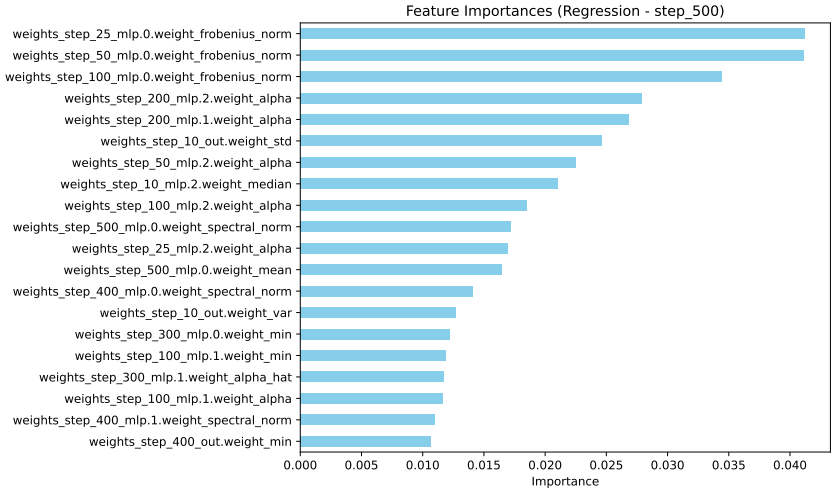
In sum, the results demonstrate that weight-based metadata can support both early-stage model triage and informed performance estimation, offering a computationally efficient alternative to full model training in time series forecasting scenarios.

4.3.1 Limitations and Future Work

While the proposed framework demonstrates promising results, several limitations constrain its generality and applicability. First, the experiments are restricted to MLPs with a fixed three-layer architecture and a maximum of 500 training steps. This design choice standardizes the experimental setting but limits the diversity of training trajectories and architectural behaviors captured by the metadata. Extending the training duration or adopting adaptive stopping criteria may reveal additional performance signals currently unobserved.

Moreover, the hyperparameter search space, though varied in learning rate, batch size, and preprocessing strategy, is not exhaustive. The meta-model is trained with default parameters and without optimization or ensembling, which may constrain its predictive accuracy.

Figure 4.7: Final ranking of feature importance for the regression task.



Future work should explore broader architectural classes, including recurrent, convolutional, and transformer-based models, which are prevalent in time series forecasting. Incorporating more expressive features to characterize the weights, such as higher-order statistics, temporal gradients, or trajectory-based measures, may further improve predictive performance. Additionally, integrating this framework into automated algorithm selection and early stopping pipelines would enhance its practical utility in real-world deployment scenarios. These extensions would improve the framework's robustness, generalization capacity, and relevance across a wider spectrum of forecasting tasks and learning configurations.

Figure 4.8: Grouped feature importance for the classification task.

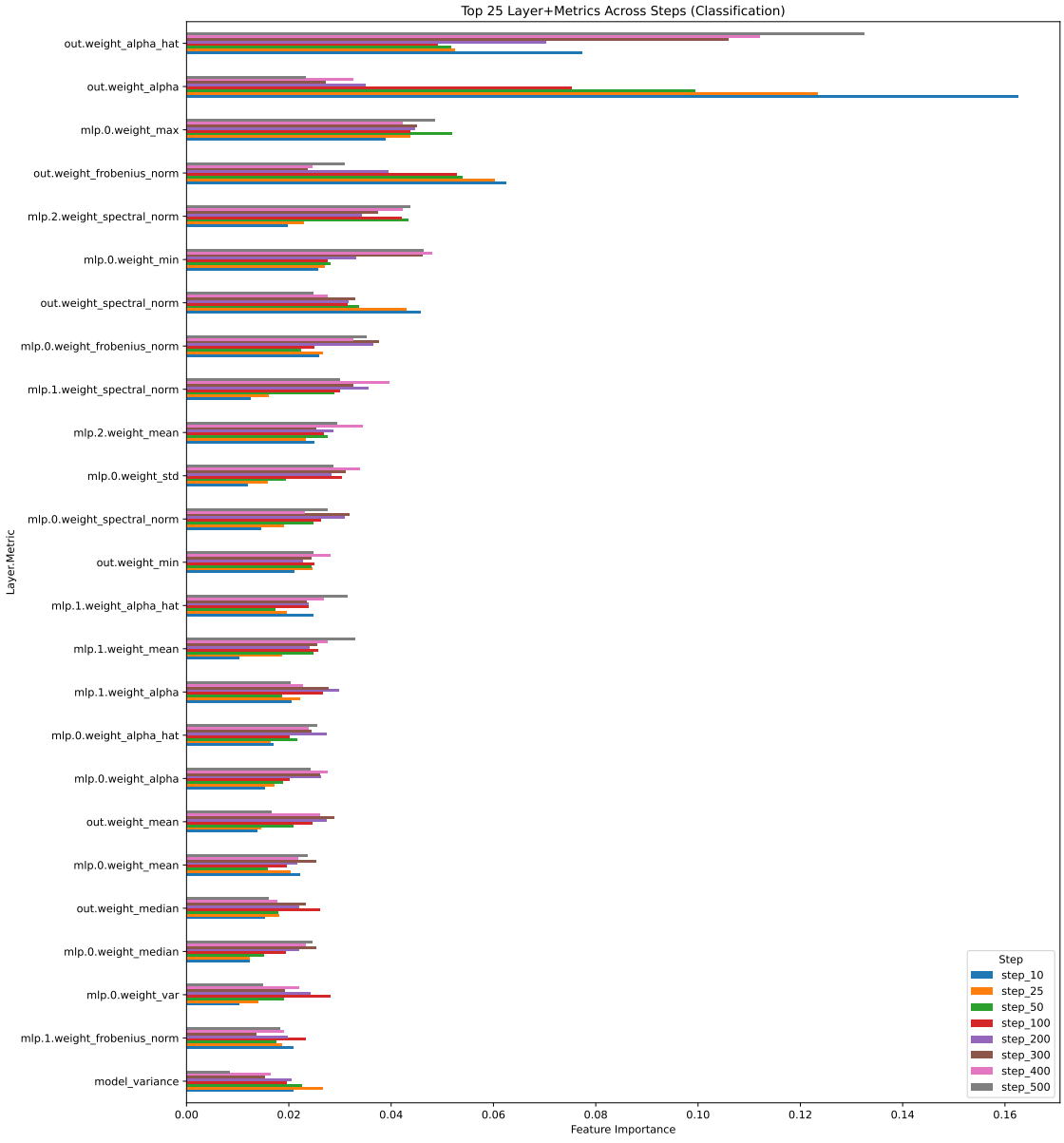
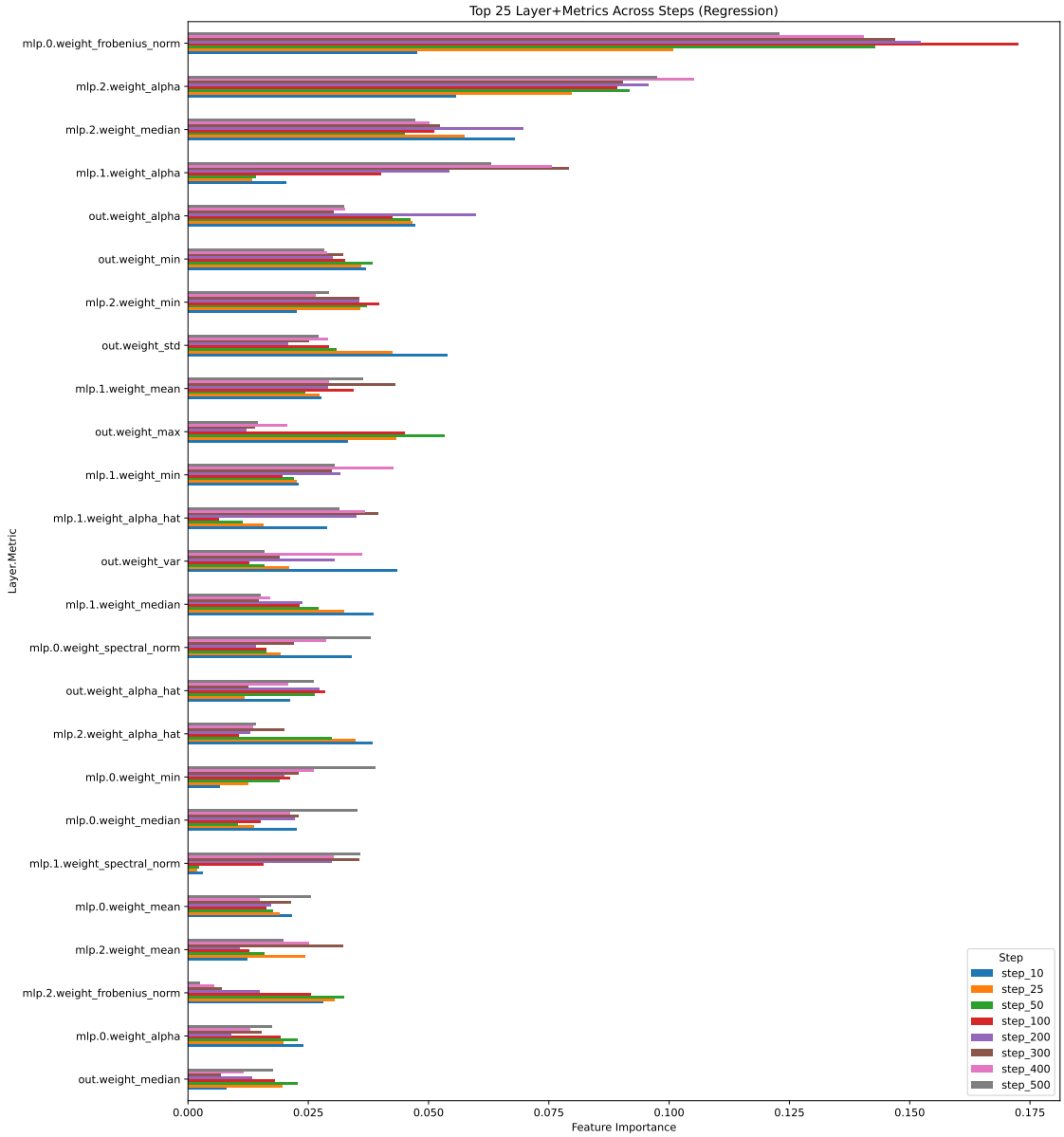


Figure 4.9: Grouped feature importance for the regression task.



Chapter 5

Conclusion

This thesis investigated the feasibility of predicting the final forecasting performance of neural networks using metadata derived from their weight configurations during training. Specifically, we proposed and evaluated a meta-learning framework designed to estimate model accuracy based on statistical properties of network weights collected at multiple training checkpoints. The motivation for this approach stems from the computational cost and resource intensity typically associated with fully training deep learning models, particularly in the context of time series forecasting.

For this approach we aim to answer certain research questions: **RQ1** explores if final model performance can be predicted, through metalearning, from weight statistics. **RQ2** investigates the earliest training stage for reliable performance predictions. And **RQ3** identifies which weight-based features are most informative for performance estimation, potentially highlighting layers with greater predictive power.

The results of **RQ1** presented in this work support the central hypothesis that weight-derived metadata encapsulates relevant information for forecasting performance prediction. The meta-models demonstrated moderate to strong predictive capabilities, especially in rank-based regression tasks. Classification models achieved performance above baseline levels, while regression models exhibited substantial rank correlation with actual forecasting errors, indicating their utility for model ranking and selection.

Temporal analysis of prediction accuracy across training stages in **RQ2** revealed that reliable estimates could be obtained at intermediate training steps, most notably around step 200, well before full convergence (step 500). This finding has direct implications for training efficiency, as it enables possible early identification and elimination of suboptimal model configurations, thereby reducing computational overhead.

Feature importance analyses, **RQ3**, further elucidated which weight-based descriptors were most informative. For classification tasks, spectral characteristics of the output layer, particularly the weighted power-law exponent, emerged as the most prominent features. Conversely, regression tasks benefited more from early-stage descriptors such as the Frobenius norm and unweighted spectral metrics in the initial hidden layers. These insights contribute to a deeper understanding of neural network training dynamics and provide guidance for future work in model interpretability.

However, the current framework presents clear limitations, such as the fixed architecture of the models used and a non-exhaustive hyperparameter search, reducing training trajectory diversity. We also do not use the trained metamodel in a practical way, limiting ourselves to its result analysis. Future research should include more diverse model architectures and integrate automated algorithm selection and early stopping to enhance applicability in a wider range of forecasting tasks.

References

[1] Agustin Garcia Asuero, Ana Sayago, and AG González. "The correlation coefficient: An overview". In: *Critical reviews in analytical chemistry* 36.1 (2006), pp. 41–59.

[2] George Athanasopoulos et al. "The tourism forecasting competition". In: *International Journal of Forecasting* 27.3 (2011), pp. 822–844.

[3] Hilan Bensusan and Alexandros Kalousis. "Estimating the predictive accuracy of a classifier". In: *European Conference on Machine Learning*. Springer. 2001, pp. 25–36.

[4] Narayan Bhusal et al. "Deep ensemble learning-based approach to real-time power system state estimation". In: *International Journal of Electrical Power & Energy Systems* 129 (2021), p. 106806.

[5] Pavel Brazdil et al. *Metalearning: Applications to data mining*. Springer science & business media, 2008.

[6] Vitor Cerqueira, Luis Roque, and Carlos Soares. "Forecasting with Deep Learning: Beyond Average of Average of Average Performance". In: *International Conference on Discovery Science*. Springer. 2024, pp. 135–149.

[7] Vitor Cerqueira, Luis Torgo, and Carlos Soares. "Model selection for time series forecasting an empirical analysis of multiple estimators". In: *Neural processing letters* 55.7 (2023), pp. 10073–10091.

[8] Cristian Challu et al. "Nhits: Neural hierarchical interpolation for time series forecasting". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 37. 2023, pp. 6989–6997.

[9] Danimir T Doncevic et al. "A recursively recurrent neural network (r2n2) architecture for learning iterative algorithms". In: *SIAM Journal on Scientific Computing* 46.2 (2024), A719–A743.

[10] Alexandros Effrosynidis, Georgios Papacharalampous, et al. "Time series and regression methods for univariate environmental forecasting: An empirical evaluation". In: *Science of The Total Environment* 901 (2023), p. 162580. DOI: 10.1016/j.scitotenv.2023.162580.

[11] Gabriel Eilertsen et al. "Classifying the classifier: dissecting the weight space of neural networks". In: *ECAI 2020*. IOS Press, 2020, pp. 1119–1126.

[12] Raquel Espinosa, Fernando Jiménez, and José Palma. "Embedded feature selection in LSTM networks with multi-objective evolutionary ensemble learning for time series forecasting". In: *arXiv preprint arXiv:2312.17517* (2023).

[13] Jean-Gabriel Gaudreault and Paula Branco. "Empirical analysis of performance assessment for imbalanced classification". In: *Machine Learning* 113.8 (2024), pp. 5533–5575.

[14] Ian J Goodfellow et al. "Multi-digit number recognition from street view imagery using deep convolutional neural networks". In: *arXiv preprint arXiv:1312.6082* (2013).

[15] Timothy M. Hospedales et al. "Meta-Learning in Neural Networks: A Survey". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2021). URL: https://doi.org/10.1109/TPAMI.2021.3079209.

[16] Rob Hyndman et al. *Forecasting with exponential smoothing: the state space approach*. Springer Science & Business Media, 2008.

[17] Rob J Hyndman. "A brief history of forecasting competitions". In: *International Journal of Forecasting* 36.1 (2020), pp. 7–14.

[18] Rob J Hyndman and George Athanasopoulos. *Forecasting: principles and practice*. OTexts, 2018.

[19] Eunjung Lee and Wonjong Rhee. "Individualized short-term electric load forecasting with deep neural network based transfer learning and meta learning". In: *IEEE Access* 9 (2021), pp. 15413–15425.

[20] Jinghang Li and Mengqi Hu. "Continuous model adaptation using online meta-learning for smart grid application". In: *IEEE Transactions on Neural Networks and Learning Systems* 32.8 (2020), pp. 3633–3642.

[21] Lisha Li et al. "Hyperband: A novel bandit-based approach to hyperparameter optimization". In: *Journal of Machine Learning Research* 18.185 (2018), pp. 1–52.

[22] Yujia Li et al. "Few-shot ultra-short-term wind power forecasting based on the meta-learning framework". In: *12th International Conference on Renewable Power Generation (RPG 2023)*. Vol. 2023. IET. 2023, pp. 296–302.

[23] Bryan Lim and Stefan Zohren. "Time-series forecasting with deep learning: a survey". In: *Philosophical Transactions of the Royal Society A* 379.2194 (2021), p. 20200209.

[24] Bryan Lim et al. "Temporal fusion transformers for interpretable multi-horizon time series forecasting". In: *International Journal of Forecasting* 37.4 (2021), pp. 1748–1764.

[25] Zimeng Lyu, Alexander Ororbia, and Travis Desell. "Online evolutionary neural architecture search for multivariate non-stationary time series forecasting". In: *Applied Soft Computing* 145 (2023), p. 110522.

[26] Spyros Makridakis and Michele Hibon. "The M3-Competition: results, conclusions and implications". In: *International journal of forecasting* 16.4 (2000), pp. 451–476.

[27] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. "M5 accuracy competition: Results, findings, and conclusions". In: *International Journal of Forecasting* 38.4 (2022), pp. 1346–1364.

[28] 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.

[29] Spyros Makridakis, Steven C Wheelwright, and Rob J Hyndman. *Forecasting: Methods and Applications*. 3rd. John Wiley & Sons, 1998.

[30] Spyros Makridakis et al. "The accuracy of extrapolation (time series) methods: Results of a forecasting competition". In: *Journal of forecasting* 1.2 (1982), pp. 111–153.

[31] Charles H Martin and Michael W Mahoney. "Implicit self-regularization in deep neural networks: Evidence from random matrix theory and implications for learning". In: *Journal of Machine Learning Research* 22.165 (2021), pp. 1–73.

[32] Charles H Martin, Tongsu Peng, and Michael W Mahoney. "Predicting trends in the quality of state-of-the-art neural networks without access to training or testing data". In: *Nature Communications* 12.1 (2021), p. 4122.

[33] Kasun Mendis, Manjusri Wickramasinghe, and Pasindu Marasinghe. "Multivariate time series forecasting: A review". In: *Proceedings of the 2024 2nd Asia Conference on Computer Vision, Image Processing and Pattern Recognition*. 2024, pp. 1–9.

[34] Pablo Montero-Manso et al. "FFORMA: Feature-Based Forecast Model Averaging". In: *International Journal of Forecasting* 36.1 (Jan. 2020), pp. 86–92. URL: https://doi.org/10.1016/j.ijforecast.2019.02.011.

[35] Hoda Nemat et al. "Blood glucose level prediction: advanced deep-ensemble learning approach". In: *IEEE journal of biomedical and health informatics* 26.6 (2022), pp. 2758–2769.

[36] Kin G. Olivares et al. *NeuralForecast: User friendly state-of-the-art neural forecasting models*. PyCon Salt Lake City, Utah, US 2022. 2022. URL: https://github.com/Nixtla/neuralforecast.

[37] Boris N Oreshkin et al. "N-BEATS: Neural basis expansion analysis for interpretable time series forecasting". In: *arXiv preprint arXiv:1905.10437* (2019).

[38] Ricardo BC Prudêncio and Teresa B Ludermir. "Meta-learning approaches to selecting time series models". In: *Neurocomputing* 61 (2004), pp. 121–137.

[39] Mohaimenul Azam Khan Raiaan et al. "A systematic review of hyperparameter optimization techniques in Convolutional Neural Networks". In: *Decision analytics journal* (2024), p. 100470.

[40] Hojjat Rakhshani et al. "Neural architecture search for time series classification". In: *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2020, pp. 1–8.

[41] Jan N. Van Rijn et al. "Fast Algorithm Selection Using Learning Curves". In: *Advances in Intelligent Data Analysis XIV*. Ed. by Elisa Fromont, Tijl De Bie, and Matthijs Van Leeuwen. Vol. 9385. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2015, pp. 298–309. URL: https://doi.org/10.1007/978-3-319-24465-5_26.

[42] Luis Roque et al. "Cherry-picking in time series forecasting: How to select datasets to make your model shine". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 39. 2025, pp. 20192–20199.

[43] Slawek Smyl. "A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting". In: *International journal of forecasting* 36.1 (2020), pp. 75–85.

[44] Thomas Unterthiner et al. "Predicting Neural Network Accuracy from Weights". In: *arXiv* (Apr. 2021). URL: http://arxiv.org/abs/2002.11448.

[45] Antonios Valkanas, André-Walter Panzini, and Mark Coates. "Towards Bayesian Learning of the Architecture, Graph and Parameters for Graph Neural Networks". In: *2022 56th Asilomar Conference on Signals, Systems, and Computers*. IEEE. 2022, pp. 852–856.

[46] Joaquin Vanschoren. "Meta-learning". In: *Automated machine learning: methods, systems, challenges* (2019), pp. 35–61.

[47] Rakshatha Vasudev et al. "Epidemic Outbreak Prediction with Ensemble of Deep Learning Models". In: *2022 IEEE 7th International Conference on Recent Advances and Innovations in Engineering (ICRAIE)*. Vol. 7. IEEE. 2022, pp. 71–76.

[48] Yuqiang Yang et al. “Short-Term Load Forecasting for Industrial Factories Based on Personalized Federated Meta-Learning”. In: *2023 IEEE 7th Conference on Energy Internet and Energy System Integration (EI2)*. IEEE. 2023, pp. 3404–3409.

[49] Han-Jia Ye et al. “A closer look at deep learning on tabular data”. In: *arXiv preprint arXiv:2407.00956* (2024).

[50] Linlin You et al. “FMGCN: Federated meta learning-augmented graph convolutional network for EV charging demand forecasting”. In: *IEEE Internet of Things Journal* (2024).

[51] Ailing Zeng et al. “Are transformers effective for time series forecasting?” In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 37. 2023, pp. 11121–11128.

[52] Lucas Zimmer, Marius Lindauer, and Frank Hutter. “Auto-Pytorch: Multi-Fidelity MetaLearning for Efficient and Robust AutoDL”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.9 (2021), pp. 3079–3090. DOI: 10.1109/TPAMI.2021.3067763. URL: https://doi.org/10.1109/TPAMI.2021.3067763.

Appendix A

Literature Review

A.1 Introduction

The systematic literature review in this chapter aims to gather and assess research on neural networks, meta-learning, predictive accuracy, and other areas where machine learning and forecasting intersect. The review seeks to deliver an overview of the research landscape, focusing on the use of meta-learning in neural networks to improve training procedures and classifier performance, as well as its application in time series forecasting problems. In order to guarantee that the review technique is transparent and replicable, descriptions of the eligibility requirements, data collection, and screening were included, creating a foundation that future researchers can build upon.

A.2 Literature Review Process

To ensure a systematic approach, a process involving the creation and usage of query strings and filtering criteria was used, maximizing relevant data collection and ensuring the inclusion of high-quality, up-to-date research on neural networks and meta-learning in the context of performance evaluation and forecasting.

A.2.1 Paper Eligibility and Collection

The biography present in the proposal [44] and records provided by the supervisors involved [3] [6] [11] [15] [34] [41] served as the starting point for literature collection. These texts were useful in identifying basic themes, problems, and specific subtopics inside neural networks and meta-learning, which helped to structure search queries for external databases. The identified keywords were selected for their relevance to the main themes associated with the problem at hand.

Subsequently, a targeted set of keywords was generated, including terms such as *Neural Networks*, *Meta-learning*, *Predictive Accuracy*, *Classifier Performance*, *Machine Learning*, *Optimization*, *Forecasting*, *Performance Evaluation* and *Generalization*. Each term was aggregated in order to simplify query creation while maintaining coverage of the main areas identified, ensuring that relevant material is captured across different databases.

A.2.2 Database Selection and Query Execution

The literature search was conducted using two major academic databases: arXiv and IEEE Xplore. These databases were selected for their extensive collections in machine learning and artificial intelligence research, offering both preprints and peer-reviewed studies essential for capturing cutting-edge developments in neural networks and meta-learning. Although other databases such as Scopus and PubMed were considered, arXiv and IEEE Xplore were chosen because of their emphasis on technological research and high relevance to the subject.

A.2.2.1 Query Strings

To ensure the retrieval of relevant material, the query strings were crafted using the main keywords identified previously, resulting in the following three main queries:

- **QS1**: “Neural networks” AND “Meta-learning” AND “Forecasting”
  - QS1 was crafted to retrieve literature focused on the application of meta-learning techniques in neural networks specifically for forecasting tasks.
- **QS2**: “Neural networks” AND “Meta-learning” AND “Predictive Accuracy”
  - QS2 focuses on research evaluating meta-learning’s impact on neural networks’ performance. In this case, "Predictive Accuracy" is included in target studies that assess how meta-learning methods enhance neural network predictions, not exactly evaluation metrics.
- **QS3**: “Neural networks” AND “Meta-learning” AND “Weight Space”
  - QS3 is designed to gather studies on the role of weight space in neural networks. Specifically, it aims to understand how the application of meta-learning techniques affects the weight space structures in order to enhance training efficiency.

The query execution yielded a total of 351 unique results across the selected databases. Table A.1 provides a breakdown of the number of results retrieved from each query and database:

Table A.1: Breakdown of results by query and database

Database	QS1	QS2	QS3	**Total**
IEEE Xplore	69	95	61	225
arXiv	28	75	23	126
Total	97	170	84	**351**

A.3 Screening and Filtering

Building on the crafted query strings, a multi-step filtering process was applied to systematically refine the initial set of 351 records, narrowing it down to the most relevant studies. Each stage of filtering is detailed below to enhance replicability and transparency.

A.3.1 Duplicate Removal

The initial step involved removing duplicate records, which reduced the total dataset to 319 unique articles. This step was critical as duplicates often appeared across multiple queries or databases, skewing the data.

A.3.2 Year of Publication

The second stage of filtering involved limiting articles by publication date, with a cutoff set for articles published from the year 2020 onward. This criterion ensured that the review focused on the most recent methodologies and findings relevant to neural networks and meta-learning, reducing the dataset to 271 records.

A.3.3 Title-Based Relevance Screening

Next, each article's title was reviewed to determine alignment with the core focus of this review. Titles were assessed for explicit mention of neural networks, meta-learning, or performance evaluation topics. This title-based relevance screening reduced the dataset to 80 records that directly addressed the intersection of neural networks, meta-learning, and classifier performance, applied to the subdomain of forecasting challenges.

A.3.4 Abstract and Conclusion Analysis

To finalize the selection, abstracts and conclusions of the remaining articles were examined for substantial insights or results that contributed to understanding performance evaluation or model generalization. Only studies that presented concrete findings or methodologies aligned with the objectives of this literature review were retained. This final screening provided the most relevant and impactful research studies, which constitute the body of literature analyzed in subsequent sections.

A.4 Replicability

This systematic review methodology was designed for ease of replication. Following each filtering step described, other researchers should achieve a similar set of literature, accounting for minor variations due to database updates or new research publications. Additionally, common exclusion reasons during each filtering phase are summarized below to further guide replicability.

Filtering Stages and Common Exclusion Reasons:

- **Title-Based Screening**: Articles were excluded if they lacked an explicit mention of neural networks, meta-learning, or relevant evaluation topics.
- **Abstract Screening**: Studies were removed if they did not contribute substantial insights to classifier performance or predictive accuracy.
- **Conclusion Review**: Articles were excluded if they lacked empirical findings relevant to neural networks and meta-learning.