

RESEARCH REPORT

---

# Time-series Forecasting: An Empirical Evaluation of the State of the Art, Ensembles- and Meta-learning Strategies

---

*Author:*

Nicolaas Pieter CAWOOD  
(2376182)

*Supervisor:*

Prof. Terence VAN ZYL



UNIVERSITY OF THE  
WITWATERSRAND,  
JOHANNESBURG

*submitted to*  
*the Faculty of Science, in fulfilment of the requirements for the degree*  
*of*  
*MSc (CW&RR) in Artificial Intelligence*  
*in the*

School of Computer Science and Applied Mathematics

January 12, 2022

# Declaration of Authorship

I, Nicolaas Pieter CAWOOD (2376182), declare that this research report titled, "Time-series Forecasting: An Empirical Evaluation of the State of the Art, Ensembles- and Meta-learning Strategies" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this research report has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this research report is entirely my own work.
- I have acknowledged all main sources of help.
- Where the research report is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

*PCawood*

---

Date: January 12, 2022

---

THE UNIVERSITY OF THE WITWATERSRAND

## *Abstract*

Faculty of Science  
School of Computer Science and Applied Mathematics

MSc (CW&RR) in Artificial Intelligence

### **Time-series Forecasting: An Empirical Evaluation of the State of the Art, Ensembles- and Meta-learning Strategies**

by Nicolaas Pieter CAWOOD (2376182)

Techniques of hybridisation and ensemble-learning are popular for improving the predictive power of forecasting methods during the past two decades. With a lack of research that combines these two promising approaches, we focus our research on the utility of the Exponential-Smoothing-Recurrent Neural Network (ES-RNN) in the pool of base models for different types of ensembles. We provide a comparison of some state of the art ensemble techniques and arithmetic model averaging as a benchmark. We experimented with the M4 forecasting competition's data set of 100,000 time-series and our results show that the Feature-based Forecast Model Averaging (FFORMA) is at average the best technique for late data fusion with the ES-RNN. However, considering the M4's Daily subset of data, stacking was the only successful ensemble at dealing with the case where all of the base model performances are similar. The experimental results lead us to conclude that model averaging is a more robust ensemble than strategies of model selection and stacking. Further, we find that gradient boosting is the superior tool for implementing ensemble-learning strategies.

## *Acknowledgements*

I would like to express my gratitude to Spyros Makridakis and his team for organising the Markidakis forecasting competitions and making the data and submissions available to the public. I would like to thank and acknowledge the work of Slawek Smyl and Pablo Montero-Manso *et al.*, for contributing their forecasting research on the ES-RNN and FFORMA. I also thank Federico Garza *et al.* for sharing their Python implementation of the FFORMA.

Finally, I thank and acknowledge my project advisor, Professor Terence van Zyl for his time, encouragement and contributions towards preparing this research report.

# Contents

<b>Declaration of Authorship</b>	<b>i</b>
<b>Abstract</b>	<b>ii</b>
<b>Acknowledgements</b>	<b>iii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Background	2
1.2 Definitions	3
1.3 Problem Statement	4
1.4 Significance and Motivation	4
1.4.1 Research Aims	5
1.4.2 Objectives	5
1.5 Research Questions	5
1.6 Delineations, Limitations and Assumptions	6
1.7 Outline	6
<b>2 Research Methodology</b>	<b>7</b>
2.1 Research design	7
2.2 The M4 forecasting competition	7
2.2.1 Performance metrics	7
2.2.2 Data description	8
Forecasting horizons	8
2.3 Overview of base models	8
2.3.1 Seasonal ARIMA	9
The autoregressive process	9
The integrated process	9
The moving average process	9
Auto ARIMA	10
Preprocessing	10
2.3.2 Theta	10
Theta's forecasting procedure	10
Preprocessing	10
2.3.3 Damped exponential smoothing	10
Preprocessing	11
2.3.4 Comb	11
Single exponential smoothing (SES)	11
Damped	11
Holt-Winters (HW)	11
Preprocessing	12

2.3.5	The ES-RNN . . . . .	12
	Preprocessing . . . . .	13
	Forecasts by the ANN . . . . .	14
	The ANN architectures . . . . .	14
	Loss function and optimiser . . . . .	14
2.4	Overview of ensembles . . . . .	15
2.4.1	Simple model averaging (AVG) . . . . .	15
2.4.2	The FFORMA . . . . .	15
	Meta-learning model . . . . .	16
	Forecasts combination . . . . .	16
	Meta-features . . . . .	17
2.4.3	The FFORMS . . . . .	17
2.4.4	Feature-weighted neural stacking (NN-STACK) . . . . .	17
	Meta-learning model . . . . .	17
	Meta-features . . . . .	17
2.4.5	Feature-weighted neural averaging (NN-AVG) . . . . .	18
	Meta-learning model . . . . .	18
	Forecasts combination . . . . .	18
	Meta-features . . . . .	18
2.5	Details of experimentation . . . . .	18
2.5.1	The evaluation protocol . . . . .	18
2.5.2	Research Instruments . . . . .	19
	Hardware and software . . . . .	19
	Data . . . . .	19
	Third-party libraries . . . . .	19
	Preprocessing of the ensembles' input data . . . . .	20
	The ensembles' architectures, hyperparameters and training . . . . .	20
2.5.3	Results data and analysis . . . . .	21
2.6	Limitations . . . . .	22
2.7	Ethical Considerations . . . . .	22
<b>3</b>	<b>Results and Discussion</b> . . . . .	<b>23</b>
3.1	Detailed results . . . . .	23
3.1.1	Observations of the Hourly subset . . . . .	25
3.1.2	Observations of the Weekly subset . . . . .	25
3.1.3	Observations of the Daily subset . . . . .	26
3.1.4	Observations of the Monthly subset . . . . .	27
3.1.5	Observations of the Yearly subset . . . . .	27
3.1.6	Observations of the Quarterly subset . . . . .	28
3.1.7	Observations of overall results . . . . .	28
<b>4</b>	<b>Conclusion</b> . . . . .	<b>31</b>
<b>A</b>	<b>The FFORMA algorithm</b> . . . . .	<b>33</b>
<b>B</b>	<b>The FFORMA meta-features</b> . . . . .	<b>34</b>

**Bibliography**

# List of Figures

3.1	OWA error distributions for the Hourly subset . . . . .	24
3.2	OWA error distributions for the Weekly subset . . . . .	25
3.3	OWA error distributions for the Daily subset . . . . .	26
3.4	OWA error distributions for the Monthly subset . . . . .	27
3.5	OWA error distributions for the Yearly subset . . . . .	28
3.6	OWA error distributions for the Quarterly subset . . . . .	29



# List of Tables

2.1	Number of series per data frequency and domain [17]. . . . .	8
2.2	The FFORMA's hyperparameters. . . . .	20
2.3	The NN-STACK hyperparameters. . . . .	21
2.4	The NN-AVG hyperparameters. . . . .	21
3.1	Average OWA performance over fifty test-folds of the M4 data set. . . . .	23
B.1	Features used in the FFORMA framework [9]. . . . .	34

# List of Abbreviations

<b>ABM</b>	<b>A</b> gent- <b>B</b> ased <b>M</b> odelling
<b>AR</b>	<b>A</b> uto <b>R</b> egressive
<b>ARIMA</b>	<b>A</b> utoregressive <b>I</b> ntegrated <b>M</b> oving <b>A</b> verage
<b>AVG</b>	<b>M</b> odel <b>A</b> veraging
<b>BMA</b>	<b>B</b> ayesian <b>M</b> odel <b>A</b> veraging
<b>ES</b>	<b>E</b> xponential <b>S</b> moothing
<b>ES-RNN</b>	<b>E</b> xponential- <b>S</b> moothing- <b>R</b> ecurrent <b>N</b> eural <b>N</b> etworks
<b>FFORMA</b>	<b>F</b> eature-based <b>F</b> orecast <b>M</b> odel <b>A</b> veraging
<b>GBM</b>	<b>G</b> radient <b>B</b> oosting <b>M</b> achines
<b>HW</b>	<b>H</b> olt <b>W</b> inters
<b>LVP</b>	<b>L</b> evel <b>V</b> ariability <b>P</b> enalty
<b>LSTM</b>	<b>L</b> ong <b>S</b> hort- <b>T</b> erm <b>M</b> emory
<b>MA</b>	<b>M</b> oving <b>A</b> verage
<b>MAE</b>	<b>M</b> ean <b>A</b> bsolute <b>E</b> rror
<b>ML</b>	<b>M</b> achine <b>L</b> earning
<b>MLE</b>	<b>M</b> aximum- <b>L</b> ikelihood <b>E</b> stimation
<b>MLP</b>	<b>M</b> ultilayer <b>P</b> erceptron
<b>MSIS</b>	<b>M</b> ean <b>S</b> caled <b>I</b> nterval <b>S</b> core
<b>NN-AVG</b>	<b>N</b> eural <b>N</b> etwork <b>M</b> odel <b>A</b> veraging
<b>NN-STACK</b>	<b>N</b> eural <b>N</b> etwork <b>M</b> odel <b>S</b> tacking
<b>OWA</b>	<b>O</b> verall <b>W</b> eighted <b>A</b> verage
<b>PF</b>	<b>P</b> oint <b>F</b> orecasts
<b>PI</b>	<b>P</b> rediction <b>I</b> ntervals
<b>QR</b>	<b>Q</b> uantile <b>R</b> egression
<b>RNN</b>	<b>R</b> ecurrent <b>N</b> eural <b>N</b> etwork
<b>RF</b>	<b>R</b> andom <b>F</b> orest
<b>SES</b>	<b>S</b> ingle <b>E</b> xponential <b>S</b> moothing
<b>SGD</b>	<b>S</b> tochastic <b>G</b> radient <b>D</b> escent
<b>SIR</b>	<b>S</b> usceptible <b>I</b> nfected, and <b>R</b> ecovered
<b>SMA</b>	<b>S</b> imple <b>M</b> odel <b>A</b> veraging
<b>XGB</b>	<b>e</b> Xtreme <b>G</b> radient <b>B</b> oosting

# List of Symbols

$\vec{x}$	Time-series
$n$	Number of time-series observations
$h$	Forecasting horizon
$\vec{w}_M$	Weightings for all models of an ensemble
$\vec{F}$	Functions that extract the meta-features ( $f$ )
$\vec{M}$	Time-series forecasting models ( $m$ )
$H$	Hessian function
$G$	Gradient function
$L$	Loss function
$\rho$	Spearman's rank correlation coefficient

# Chapter 1

## Introduction

Although it is a notoriously tricky task, the forecasting of time-series is nevertheless an important one that boasts numerous research efforts over the years. Markidakis *et al.* [1] emphasise two facts about the field: first, no one has prophetic powers to predict the future accurately, and second, all predictions are subject to uncertainty, especially within social contexts.

The term "forecasts" were first coined nearly two hundred years ago by FitzRoy, a meteorologist who made daily weather predictions after considering the value of predicting coming storms might hold for sailors at sea. This fueled the growing realisation that weather does follow predictable patterns over large areas [2]. Forecasting's potential has subsequently been realised for any domain within the physical sciences; moreover, our understanding of the sciences grew, along with increased access to large amounts of information. In the nineteenth century, Yule [3] made a significant contribution by proposing that every time-series can be thought of as the realisation of a stochastic process. This very concept led to the rise of several prime statistical time-series techniques like the autoregressive (AR) and moving average (MA) models [4]. Nowadays, most forecasting literature implements artificial neural networks (ANN) or combines them with traditional models. Zhang *et al.* [5] identify this attraction to ANNs as their ability to self-adapt and model underlying relationships within the data, even when unknown.

However, Markidakis *et al.* [1] point out that machine learning models deliver poor performances for the forecasting of social sciences compared to the naive benchmark: on the seasonally adjusted data (Naïve model 2). They emphasise that we have a limited theoretical foundation of their underlying mechanisms, and despite having large amounts of data available, we usually want to forecast a series that is implicated by unknown factors. For this reason, it is important for initiatives to drive the continuous research of improved forecasting methods. The Makridakis Competitions <sup>1</sup> (or M-Competitions) are undoubtedly the most influencing such an initiative, and they are a standard benchmark for establishing innovative forecasting methods [1].

There exist many techniques underpinned by different principles to perform forecasting, and numerous forecasting studies have shown that combination models generally outperform individual ones [6]–[8]. Given the top entries of the M4 competition (i.e., the FFORMA [9] and ES-RNN [10]) - state

---

<sup>1</sup><https://mofc.unic.ac.cy/>

of the art forecasting approaches adopt ensemble learning and hybridisation to improve their predictive performance. This comes as no surprise, as ensembles are known to outperform their constituent models for many machine learning problems [11]. Likewise, the integration of models with different architectures has been reported to deliver superior results [12], [13]. For this reason, we focus our research on combination strategies rather than individual forecast methods, and the reader interested in a comprehensive study of statistical, and machine learning (ML) forecast models are referred to a large comparative paper by Makridakis *et al.* [14].

From a review of contemporary forecasting literature [9], [10], [15], we identify the data set from the M4 forecasting competition as the most prominently used, and thus the most relevant data set for our research. The data is comprised of 100,000 time-series of different seasonality from the micro, macro, industry, finance, demographic and other domains.

## 1.1 Background

Data science/ML techniques are currently the preferred family for research of forecasting time-series, owing to their predictive accuracy [16]. Makridakis *et al.* [17] recently presented a large scale comparative study of 100,000 time-series and 61 forecasting methods. They compare the accuracy of statistical methods (e.g., Autoregressive integrated moving average (ARIMA), Holt-Winters (HW), and others) versus that of ML ones (e.g., Multilayer Perceptron (MLP), recurrent neural network (RNN) and others.) Furthermore, similar to others [6]–[8], they find that utilising models of both kinds in combination models produce the best results.

Despite the immense diversity of the forecasting models, the no free lunch theorem [18] holds that no single algorithm universally outperforms any other for all problems. This raises an important question of how to choose the best algorithm for a given time-series. Meta-learning was originally proposed by Arinze [19] to rank time-series forecasting models using decision trees to recommend a single model among six others based on their learned performance for six features describing the series (e.g., autocorrelation, trend, coefficients.) Raftery *et al.* [20] point out, however, that single model selection is open to issues of model uncertainty and proposed using Bayesian model averaging (BMA) to make weighted average predictions as the posterior probability of each model being correct given the same data set. Stacking is another ensemble method that uses a combiner model to learn from the predictions of multiple methods [21] using cross validation, instead of using posterior probabilities. Clarke [21] reported stacking as a more robust method than BMA for computations that involve sensitive changes over their variables. Ribeiro and Coelhoa [22] studied contemporary ensemble methods in forecasting time-series within agribusiness. In their work, they compare approaches of bagging with random forests (RF), boosting with gradient boosting machines (GBM), extreme gradient boosting machine (XGB)

and a stacked generalisation of machine learning models. Their findings suggest that gradient boosting methods generally produce the lowest prediction errors.

Ensemble methods' performance might also be improved by utilising additional time-series statistics as meta-features in the learning process. Ca-wood and van Zyl [23] for example proposed a feature-weighted stacking approach that improves the regression from weak learner predictions by supplementing them with time-series statistics that are extracted from the input series. Some other recent work [9], [24], [25] has established many valuable statistics for meta-learning, including, but not limited to the linearity, curvature, autocorrelation, stability and entropy.

The top two submissions of the M4 forecasting competition both implement ensemble learning to produce state of the art results. The runner-up: FFORMA framework [9] implements an XGB meta-learning model that learns the model weightings of the ensemble based on their performance learned for several meta-features extracted from the input series. The weightings previously learned by the trained meta-model are then used to combine the models predictions as a feature-weighted average.

The Exponential Smoothing-Recurrent Neural Network [10] (ES-RNN) is a hybrid forecasting model, and winning submission of the M4 competition, that merges a modified Holt-Winters and dilated long short-term memory (LSTM) stacks. Hybrid methods integrate linear- and nonlinear models as time-series, in reality, are rarely pure linear or nonlinear [26]. Within the context of time-series forecasting, the idea of hybrid machine learning models was first proposed by Zhang [26], who proved that the hybridisation of the ARIMA and MLP models yield improved accuracy since it takes advantage of combining the strength of both linear- and nonlinear models. The authors note that the main advantage of neural networks is their ability to do nonlinear modelling. In their implementation, the MLP learns from the residuals of the linear ARIMA method to make the final integrated predictions. Hybrid methods have gained traction in recent time-series forecasting works [27]–[29]. The most common approach is to combine deep learning with a linear method, similar to the initial work by Zhang. Some [30], [31] have proposed the hybridisation of fuzzy logic techniques with ML models. Others [32], [33] have also reported success for integrating ML models with genetic algorithms like particle swarm- and ant colony optimisation.

## 1.2 Definitions

The following key terms are defined within the context of this research and they are intended to help the reader understand the study better.

**Base model/weak learner** A single statistical or machine learning forecast method.

**Characteristic statistics/meta-features** Endogenous features that are extracted from the input series that statistically describe the complexity and behaviour thereof.

**Ensemble learning** The combination of multiple algorithms' predictions to obtain better predictive performance.

**Hybridisation** The integration of two or more distinctively powerful techniques to blend their unique advantages to deliver a single- and more powerful method that encapsulates the advantages of each of its constituent models.

**Meta-learner/strong learner** The core learner of the ensemble that learns from both the metadata and the predictions of two or more base models.

**Stacking** A stacked generalisation of base models that combines their predictions.

### 1.3 Problem Statement

Techniques of model integration and ensembling have shown to improve forecast models' predictive power in the literature. Consequently, the combination of these two concepts has delivered state of the art results, as noted for the winning submission for the M4 competition: the ES-RNN. Furthermore, given the competition ranking of the FFORMA framework, the adoption of a feature weighted meta-learning strategy has shown to be the most robust combination technique amongst numerous others.

However, the pool of base models from the FFORMA's implementation were traditional statistical models, and to the best of our knowledge, there is no study that researches how well this ensemble would perform when an advanced model like the ES-RNN is added to the pool of base models. Furthermore, there is a gap in the literature that evaluates the performance of these techniques against other ensembles, using the forecasts from the same pool of forecasting models. To this end, we propose implementing and evaluating different forecast models and ensembles to empirically determine whether we can single out any of them as the state of the art forecasting technique.

### 1.4 Significance and Motivation

Makridakis *et al.*[1] point out that hybrid machine learning models integrated with classical time-series forecast models currently perform at a conceptual level. This is evident when considering the performance of the ES-RNN model in the M4 forecasting competition. However, the literature has shown that a combination of models generally improves the predictions from standalone forecasting models. Therefore, it appears that researching different ensembling techniques that include hybrid machine learning base models is the direction to take for forecasting.

We propose using the forecasts of selected submissions of the M4 forecasting competition and use them as our weak learner forecasts for different ensemble techniques (model selection, model averaging and stacking). We

optimistically expect our findings to provide future researchers with empirical evidence of a state of the art benchmark for their work.

### 1.4.1 Research Aims

The stochastic environments of social contexts make the forecasting thereof tricky. Nevertheless, maximising the accuracy of our forecasting models are of great importance as they offer countless benefits. These benefits are sought after in industries, hence why it is typical for literature on forecasting some business implicated time-series [34], [35].

This research aims to combine state of the art techniques for data and model fusion to forecast time-series. Furthermore, we aim to provide a comparison of our results with that of the top submissions of the M4 forecasting competition, to answer whether we might single out any method as the state of the art approach to forecast time-series from a large data set of 100,000 time-series of different domains and seasonality.

### 1.4.2 Objectives

The objectives of this research are as follows:

- To survey the methodologies of the top-two submissions for the M4 forecasting competition and analyse how we might combine their key advantages of hybridisation and feature weighted model averaging.
- To evaluate the superiority of ensembles in forecasting and compare the performance of the ES-RNN and FFORMA against other ensemble techniques.
- Another objective is to deliver a study that compares multiple ensembles using the same data produced by a common pool of forecasting models.
- Our primary objective is to determine whether we can single out any method as the state of the art forecasting technique for future research efforts.

## 1.5 Research Questions

The questions this researches proposes to answer are as follows:

1. How does the accuracy of a state of the art forecasting method, the ES-RNN, compare to that of ensembles that utilise the ES-RNN as a weak learner?
2. How does the accuracy of different ensemble approaches (model stacking, model averaging and model selection) compare against each other?



## 1.6 Delineations, Limitations and Assumptions

Although the M4 competition has an optional prediction interval forecasting component, this research is limited to point forecasts only. We only consider the M4 competition's data set as it has the best potential for comparing our results with the literature. As a result, the outcomes might not apply to other data sets or problems other than time-series forecasting. We do not implement all of the base models discussed in this paper, and instead we reuse the original forecasts from the M4 submissions as the data for our ensembles instead.

## 1.7 Outline

The remainder of this paper is organised as follows: Chapter 2 proposes the methodology that this research follows. First, we discuss the data sets and requirements of the M4 forecasting competition. We then provide overviews of the base models and ensembles' methodologies, which are key instruments to our experimentation. The chapter's final sections report the evaluation protocol and other important research instruments of our experimentation. In Chapter 3, we provide our findings as well as a discussion of forecasting performances. Finally, Chapter 4 concludes the findings of this work, and we make suggestions for future research efforts of forecasting and ensemble techniques.

## Chapter 2

# Research Methodology

### 2.1 Research design

We undertake a confirmatory experimental research design where we experimentally determine how effective some ensembles are. We reuse the forecasts of some of the M4 competition's submissions as the inputs of our ensembles to determine by how much the accuracy of the ES-RNN might have been boosted with an ensemble-learning strategy. We would eventually test and support the hypothesis that a feature-weighted ensemble should provide state of the art forecasting performance.

### 2.2 The M4 forecasting competition

#### 2.2.1 Performance metrics

The rules of the competition are available on the M4 competition's website<sup>1</sup>. The participants had to submit point forecasts (PFs) and, optionally, prediction intervals (PIs) for 100,000 series.

The overall weighted average (OWA) computes the average of the two most popular accuracy measures to evaluate the performance of the PFs relative to the Naïve model 2. The OWA, therefore, combines the two metrics of the symmetric mean absolute percentage error (sMAPE) [36] and mean absolute scaled error (MASE) [37], each calculated as follows:

$$sMAPE = \frac{1}{h} \sum_{t=1}^h \frac{2|Y_t - \hat{Y}_t|}{|Y_t| + |\hat{Y}_t|} \times 100\% \quad (2.1)$$

$$MASE = \frac{1}{h} \frac{\sum_{t=n+1}^{n+h} |Y_t - \hat{Y}_t|}{\frac{1}{n-m} \sum_{t=m+1}^n |Y_t - Y_{t-m}|} \quad (2.2)$$

where  $Y_t$  is the actual value of the time-series,  $\hat{Y}_t$  is the predicted value of the time-series,  $h$  the length of the forecasting horizon,  $n$  is the number of in-sample data points.  $m$  defines the time interval between each successive observation, i.e., 12 for time-series that have a monthly frequency, 4 for those with a quarterly frequency, 24 for hourly series and 1 for the other frequencies

---

<sup>1</sup><https://usermanual.wiki/Document/M4CompetitorsGuide.1491768831/html>

that are nonseasonal series. The OWA error is then computed as the average of the MASE and sMAPE errors relative to the Naïve 2 predictions as follows:

$$OWA = \frac{1}{2} \left( \frac{MASE}{MASE_{Naive_2}} + \frac{sMAPE}{sMAPE_{Naive_2}} \right) \quad (2.3)$$

### 2.2.2 Data description

The M4 data set contains 100,000 time-series of different frequencies, i.e., yearly, quarterly, monthly, weekly, daily and hourly. The minimum number of observations varies for the different subsets, e.g., 13 for the yearly series and 16 for quarterly. The data is freely available on Github<sup>2</sup> and Table 2.1 provides a summary of the number of series per frequency and domain. Economic, Finance, Demographics, and Industries are among the domains, with data from Tourism, Trade, Labor and Wage, Real Estate, Transportation, Natural Resources, and the Environment also included.

TABLE 2.1: Number of series per data frequency and domain [17].

Data subset	Micro	Industry	Macro	Finance	Demographic	Other	Total
Yearly	6,538	3,716	3,903	6,519	1,088	1,236	23,000
Quarterly	6,020	4,637	5,315	5,305	1,858	865	24,000
Monthly	10,975	10,017	10,016	10,987	5,728	277	48,000
Weekly	112	6	41	164	24	12	359
Daily	1,476	422	127	1,559	10	633	4,227
Hourly	0	0	0	0	0	414	414
Total	25,121	18,798	19,402	24,534	8,708	3,437	100,000

Redd *et al.* [38] describe the data as univariate, with no temporal links between the series, making it difficult to forecast because the series is the only representation of the signal.

### Forecasting horizons

The length of the forecasts for each frequency are 6 forecasts for yearly data, 8 forecasts for quarterly data, 18 forecasts for monthly data, 13 forecasts for weekly data, 14 forecasts for daily data, and 48 forecasts for hourly data.

## 2.3 Overview of base models

The combination of traditional statistical models and deep learning models have shown to produce promising results [23]. Consequently, we chose four statistical methods, among which two of them were the best methods for previous Makridakis forecasting competitions. Lastly, we experiment with

<sup>2</sup><https://github.com/Mcompetitions/M4-methods/tree/master/Dataset>

the ES-RNN as a hybrid deep learning model and the winner of the M4 forecasting competition. This section gives an overview of each model used in our experimentation.

### 2.3.1 Seasonal ARIMA

The ARIMA method is a standard method for the comparison of forecasting method's performance. Fattah *et al.* [39] describe the ARIMA an iterative approach with three sub-processes. The seasonal ARIMA model might be written as follows:

$$\text{ARIMA}(p, d, q)(P, D, Q)_m \quad (2.4)$$

where  $p$  denotes the number of autoregressive terms,  $d$  denotes the number of differences  $q$  is the number of moving averages. The seasonal terms (P, D and Q) are similar to the nonseasonal ones, but the timesteps are defined by  $m$ - the frequency of the seasonality .

#### The autoregressive process

The time-series is assumed as linear and it might be found from the linear function:

$$Y_t = \alpha_1 Y_{t-1} + \epsilon_t \quad (2.5)$$

where  $\alpha_1$  is the self-regression coefficient and  $\epsilon$  is a random shock component on each observation.

#### The integrated process

For a time-series to become stationary, the integrated process represents the differencing of observations. (i.e., observations are replaced with the difference between themselves with previous observations.) The integrated process is simply defined as:

$$Y_t = Y_{t-1} + \epsilon_t \quad (2.6)$$

where  $\epsilon$  is white noise in the signal.

#### The moving average process

The moving average term represents the past error and might be found using the following equation:

$$Y_t = \epsilon_t - \theta_1 \epsilon_{t-1} \quad (2.7)$$

where  $\theta_1$  is the coefficient to be multiplied to the previous error.

### Auto ARIMA

This implementation automates the selection of the ARIMA parameters, using the auto-ARIMA method [40] that uses maximum-likelihood estimation (MLE) to approximate the ARIMA parameters.

### Preprocessing

The ARIMA is intended to model stationary time-series and thus it is common to remove the trend and stabilise the variance as before the model is fit.

### 2.3.2 Theta

The Theta-model [41] was the best method of the M3 forecasting competition. Theta is a simple forecasting method that averages the extrapolated Theta-lines, computed from two given Theta-coefficients, applied to the second differences of the time-series:

$$X''_{data} = X_t - 2X_{t-1} + X_{t-2} \quad (2.8)$$

$$\wedge X''_{new} = \theta \cdot X''_{data} \quad (2.9)$$

The first Theta-line ( $\theta = 0$ ) describes the time-series through a linear trend and it is extrapolated using linear regression. The second Theta-line (usually  $\theta = 2$ ) is extrapolated using simple exponent smoothing (SES) [42].

### Theta's forecasting procedure

The time-series are firstly decomposed into the two Theta-lines using equation 2.9. The extrapolation procedures mentioned above are then applied and their forecasts are averaged to produce a combined forecast.

### Preprocessing

A 90% autocorrelation test is adopted to determine whether the data are seasonal, and when it is the case, the data are seasonally adjusted by applying classical multiplicative decomposition.

### 2.3.3 Damped exponential smoothing

Holt's [43] linear trend method extended exponential smoothing to do forecasting of data that includes a trend component. McKenzie and Gardner [44] describes the Damped method as a modification of Holt's method that adds an autoregressive-damping parameter  $\phi$  imposed on the trend component.

The model is thus found using the following additive equations:

$$y_t = l_{t-1} + \phi b_{t-1} + \epsilon_t \quad (2.10)$$

$$l_t = l_{t-1} + \phi b_{t-1} + (1 - \alpha)\epsilon_t \quad (2.11)$$

$$b_t = \phi b_{t-1} + (1 - \beta)\epsilon_t \quad (2.12)$$

where  $y_t$  is the time-series observed values,  $l_t$  denotes the series' level,  $b_t$  is the gradient of the linear trend and  $\epsilon_t$  is a single source of error.

### Preprocessing

The same preprocessing is applied as per the Theta model.

#### 2.3.4 Comb

Comb (or COMB S-H-D) is the arithmetic average of the three exponential smoothing methods: Single, Holt-Winters and Damped exponential smoothing [45]. Comb was the winning approach for the M2 forecasting competition and it was used as a benchmark for evaluating all other methods in the M4 forecasting competition.

#### Single exponential smoothing (SES)

Nazim and Afthanorhan [46] mentioned that SES is used to forecast time-series that are nonseasonal and without a trend. They give the general equation for SES as:

$$F_{t+m} = \alpha y_t + (1 - \alpha)F_t \quad (2.13)$$

where  $F_{t+m}$  is the single exponential smoothed value at the  $m$ -step-ahead period.  $y_t$  is the actual value at time step  $t$ ,  $\alpha$  is a smoothing constant between 0 and 1, and  $F_t$  is the forecast at time step  $t$ .

#### Damped

The same model discussed in Section 2.3.3.

#### Holt-Winters (HW)

In contrast to SES, the HW exponential smoothing was proposed for data that are seasonal and exhibits a trend [47]. Kalekar [47] mentions that there are two HW models for different types of seasonality, namely the multiplicative and additive seasonal model.

**The Multiplicative Seasonal Model** is appropriate for time-series with a seasonal pattern that's amplitude is proportional to the average level of the

series. The time-series is modelled using the following notation:

$$y_t = (b_t + b_{2t})S_t + \epsilon_t \quad (2.14)$$

where  $b_1$  is the base signal,  $b_2$  is an optional linear trend component,  $S_t$  is a multiplicative seasonal factor and  $\epsilon_t$  is the random error component.

**The Additive Seasonal Model** is appropriate for time-series with a seasonal pattern that's amplitude is independent of the average level of the series. The time-series is modelled using the following notation:

$$y_t = b_t + b_{2t} + S_t + \epsilon_t \quad (2.15)$$

where  $b_1$  is the base signal,  $b_2$  is an optional linear trend component,  $S_t$  is a multiplicative seasonal factor and  $\epsilon_t$  is the random error component.

## Preprocessing

Seasonal adjustments are taken into account according to the Theta model.

### 2.3.5 The ES-RNN

The ES-RNN method performs the fusion of ES models with LSTM networks to produce more accurate forecasts than those predicted by either statistical or ML approaches. Smyl [10] describes the three main elements of the ES-RNN approach as deseasonalisation and adaptive normalisation, generation of forecasts and ensembling.

The seasonality patterns of the time-series provided for the M4 competition are diversified as they are in the format of numeric vectors without timestamps, and they originated from various sources. The first element of Smyl's approach normalises and deseasonalises the series on the fly by ES decomposition to deal with this. The ES decomposition also computes the level, seasonality, and second seasonality components to integrate with the RNN forecasts.

Smyl's methodology allocates the predictions of the time-series' trends to the RNN method since it can model nonlinear relationships. It is a common approach for hybrid forecasting models to exploit the benefits of linear and nonlinear methods by integrating them. The second element is a dilated LSTM network [48], which takes the deseasonalised and normalised data to discover the potential nonlinear relationships within the data.

The final element is the ensembling of the forecasts from multiple instances of the hybrid model achieved by the first two steps. The ensembling helps mitigate the model and parameter uncertainty [49] and provides the advantage of averaging over model combinations [50]. In Smyl's methodology, the hybrid instances are trained with the same architecture for independent randomly initialised runs and different subsets of the time-series if

computationally feasible. The final forecast for a given series is then the average of the forecasts produced by the top-N best models.

### Preprocessing

Since the M4 competition's data are from various sources, it contains different seasonality patterns, even when categorised under the same frequency.

In Smyl's methodology, deseasonalisation is achieved on the fly using ES models [51]–[53] for time-series that are nonseasonal (yearly and daily frequencies), single-seasonal (monthly, quarterly and weekly frequencies) and double-seasonal (hourly frequency.) In addition, the formulas are updated to remove the linear trend component, which is modelled using the ANN instead. The updated formulas are as follows:

Nonseasonal models:

$$l_t = \alpha y_t + (1 - \alpha)l_{t-1} \quad (2.16)$$

Single-seasonality models:

$$\begin{aligned} l_t &= \alpha y_t / s_t + (1 - \alpha)l_{t-1} \\ s_{t+K} &= \beta y_t / l_t + (1 - \beta)s_t \end{aligned} \quad (2.17)$$

Double-seasonality models:

$$\begin{aligned} l_t &= \alpha y_t / (s_t u_t) + (1 - \alpha)l_{t-1} \\ s_{t+K} &= \beta y_t / (l_t u_t) + (1 - \beta)s_t \\ u_{t+L} &= \gamma y_t / (l_t s_t) + (1 - \gamma)u_t \end{aligned} \quad (2.18)$$

where  $y_t$  is the value of the series at time step  $t$ ;  $l_t$ ,  $s_t$  and  $u_t$  are the level, seasonality and second-seasonality components respectively;  $K$  denotes the the number of seasonal observations (i.e., four for quarterly, 12 for monthly and 52 for weekly;) and  $L$  is the number of double-seasonal observations (168 for the hourly frequency data.)

Smyl adopts normalisation using the typical approach of constant size, rolling input and output windows to normalise the level and seasonality components produced by the ES methods in 2.16 - 2.18. The input window size is defined after experimentation, and the size of the output window is always equal to the length of the forecasting horizon. The values of the input and output windows are then divided by the last value of the level of the input window and, if the series is seasonal, additionally divided by the seasonality component. Lastly, the log function is applied to counter the effects of outliers on the learning process, and the values are thus squashed to the logarithmic scale.

Furthermore, the time-series' domain (e.g., micro, macro and finance) are one-hot encoded and presented as the only meta-features to the RNN model.



### Forecasts by the ANN

The normalised, deseasononaised and squashed values of level and seasonality and the meta-features are presented as inputs to the ANN. The outputs of the ANN are then integrated to complete the ES-RNN hybridisation in the following way:

Nonseasonal models:

$$\hat{y}_{t+1,\dots,t+h} = \exp(NN(x)) * l_t \quad (2.19)$$

Single-seasonal models:

$$\hat{y}_{t+1,\dots,t+h} = \exp(NN(x)) * l_t * s_{t+1,\dots,t+h} \quad (2.20)$$

Double-seasonal models:

$$\hat{y}_{t+1,\dots,t+h} = \exp(NN(x)) * l_t * s_{t+1,\dots,t+h} * u_{t+1,\dots,t+h} \quad (2.21)$$

where  $x$  is a vector of the preprocessed data discussed above,  $NN(x)$  is a vector output from the NN and  $h$  is the forecasting horizon. The  $l$ ,  $s$  and  $u$  components are from the outputs of the ES models during the preprocessing step.

### The ANN architectures

Although the ES-RNN's architectures are different for each of the six data subsets, we review some of their key components.

The ANNs of the ES-RNN are constructed from dilated LSTM [48] stacks, and in some cases, followed by a nonlinear layer and always a final linear layer. Smyl refers to the linear layer as the "adapter" layer since it adapts the size of the last layer to the size of the forecasting horizon, or twice the size of the forecasting horizon for prediction interval (PI) models.

DilatedRNNs improve regular RNNs' performance with significantly fewer parameters, owing to the dilated recurrent skip connections [48]. Smyl also extends the dilated LSTM with an attention mechanism that exposes the hidden states to the weights of the previous states- a horizon equal to the dilation. To achieve this, Smyl embeds a linear two-layer network into the LSTM.

### Loss function and optimiser

The ES-RNN implements a pinball loss function to fit the models using Stochastic Gradient Descent (SGD.) The loss is defined as follows:

$$\begin{aligned} L_t &= (y_t - \hat{y}_t)\tau, \text{ if } y_t \geq \hat{y}_t \\ &= (\hat{y}_t - y_t)(1 - \tau), \text{ if } \hat{y}_t > y_t \end{aligned} \quad (2.22)$$

where  $\tau$  is configured typically between 0.45 and 0.49. Smyl notes that the pinball function is asymmetric and that it penalises values outside a quantile

range differently to deal with any biasing, and minimising it produces quantile regression.

A level variability penalty (LVP) is implemented as a regulariser to perform smoothing of the level values in the loss function. Smyl notes that this drastically improves the performance of the ANN as it can concentrate on modelling the trend instead of over-fitting seasonality-related patterns. The ES-RNN's LVP is found as follows:

- Compute the log change, i.e.,  $d_t = \log(y_{t+1}/y_t)$
- Compute the difference of the changes:  $e_t = d_{t+1} - d_t$
- Square and average the differences

Lastly, this penalty is multiplied by a constant parameter in the range of 50 - 100 before adding it to the loss functions.

## 2.4 Overview of ensembles

We experiment with three different kinds of ensemble techniques, namely, model stacking, model averaging and model selection. This section gives an overview of each ensembling technique used in our experiments that are different in their complexity and meta-learning strategies.

### 2.4.1 Simple model averaging (AVG)

As a simple benchmark, we implement a model averaging [54] technique that averages the forecasts of the weak learners. The combined forecast for a set of forecast models  $M$ , might thus be expressed by the following notation:

$$\hat{y}_t = \frac{1}{n} \cdot \sum_{m=1}^n y_{m_t} \quad (2.23)$$

where  $n$  is the number of models in  $M$  and  $y_{m_t}$  is the forecast of a single model at timestep  $t$ .

### 2.4.2 The FFORMA

The FFORMA framework adopts a feature-weighted model averaging strategy, that has a meta-learner learn how effective a pool of forecasting models are at their task for different regions of the meta-feature space and then combine their predictions based on the learned model weightings. In addition, the FFORMA implements the gradient tree boosting model of XGBoost [55], which the authors note is computationally efficient and has shown promising results for other problems.

Montero-Manso *et al.*'s [9] meta-learning methodology use inputs of nine models and features extracted from the time-series, which measures the characteristics of a time-series: including, but not limited to, features of lag, correlation, the strength of seasonality and spectral entropy.

### Meta-learning model

Montero-Manso *et al.* implement a custom objective function for the XGBoost to minimise. XGBoost requires both a gradient and hessian of the objective function to fit the model. The functions for their model is derived as follows.

$p(f_n)m$  is firstly defined as the output of the meta-learner for model  $m$ . A soft-max transformation is applied to the numeric values to compute the model weights as the probability of each model being the best as follows:

$$w(f_n)m = \frac{\exp(p(f_n)m)}{\sum_m \exp(p(f_n)m)} \quad (2.24)$$

For each  $n$  time-series, the contribution of each method for the OWA error is denoted as  $L_{nm}$ . The weighted average loss function is then computed as follows:

$$\bar{L}_n = \sum_{m=1}^M w(f_n)m L_{nm} \quad (2.25)$$

The gradient of the loss function is then computed as follows:

$$\bar{G}_{nm} = \frac{\partial \bar{L}_n}{\partial p(f_n)m} = w_{nm} = (L_{nm} - \bar{L}_n) \quad (2.26)$$

The hessian of the objective function is then finally derived as follows:

$$\bar{H}_{nm} = \frac{\partial \bar{G}_n}{\partial p(f_n)m} \approx \hat{H}_t = w_n(L_n(1 - w_n) - G_n) \quad (2.27)$$

In order to minimize the objective function  $\bar{L}$ , the functions  $G$  and  $\hat{H}$  are passed to XGBoost, and the model's hyperparameters are found using Bayesian optimisation on a limited search space that is determined based on preliminary results and rules-of-thumb.

### Forecasts combination

Montero-Manso *et al.*'s methodology combine the predictions of their pool of forecasting models using algorithm 1 (see Appendix A). The forecasts are produced from the meta-learner's estimated model weightings, given the meta-features of a new time-series. Thus, the combination is achieved by using  $w(f_x)$ , an  $M$ -vector of weights, with the forecasts of a set of  $M$  forecasting models for each time-series  $x$ .  $f$  is a vector of meta-features that are computed by a set of  $F$  functions for calculating the time-series features.

## Meta-features

The time-series features used in the FFORMA are extracted using the R package called *tsfeatures* [56]. Table B.1 (see Appendix B) provides brief descriptions of the features used.

For non-seasonal time-series, features that only apply to seasonal time-series are set to zero. Unlike the ES-RNN, the domain-specific features supplied with the M4 data set are not utilised in the FFORMA.

### 2.4.3 The FFORMS

The FFORMA was built on the Feature-based FORecast Model Selection (FFORMS) [57] framework that learns to select a single model from a pool of forecasting models according to their varying performance observed over some meta-data feature space.

A random forest ensemble learner [58] is adopted by the original FFORMS implementation to classify a single model as the most relevant for some time-series features extracted from the reference series. In this paper however, we instead replace the weighted-averaging of the FFORMA, with the selection of the model with the highest allocated weighting. We highlight this change in algorithm 1 (see Appendix A).

### 2.4.4 Feature-weighted neural stacking (NN-STACK)

Recently, Cawood and van Zyl [23] proposed model stacking to do forecasting of nonseasonal time-series. Their implementation performs regression over a feature space consisting of both the model forecasts and a set of statistics that are extracted from the time-series. This paper builds on their work by experimenting with more meta-features and a larger data set, of different domains and seasonality.

## Meta-learning model

We thus combine the forecasts of a single timestep, performing regression over the model forecasts and the meta-features extracted from the input series. The MLP model performs regression over inputs of the model forecasts and the extracted meta-features, with the target variables as the predicted series' actual values.

A perceptron of a basic architecture is adopted, and each layer is transformed using the ReLU activation function. The model is fit using mini-batches and the Adam [59] stochastic gradient descent algorithm with a mean absolute error (MAE) loss function.

## Meta-features

Although only four statistics were used with this approach before, we propose experimenting with the larger set of features used by the FFORMA framework.

Feature selection is achieved using Spearman's rank correlation coefficient ( $\rho$ ) [60], that measures the extent of the meta-features' correlation with the change in each model performance. This procedure helps reduce problems with overfitting, and only the most corellating features are included in the ensemble.

### 2.4.5 Feature-weighted neural averaging (NN-AVG)

We additionally propose a MLP meta-learner, that takes the time-series statistics as inputs and a one-hot encoded vector of the best performing model as the model's targets. This is a deep-learning approach to the FFORMA's methodology that adopts a SoftMax activation function in the output layer to normalise the network's output to a probability distribution of the base models' predicted adequacy to model a time-series.

#### Meta-learning model

The neural network is of a deep architecture and adopts a ReLU activation function at each layer except for the SoftMax activated output layer. The model is fit using mini-batches and the Adam stochastic gradient descent algorithm with a categorical cross-entropy loss function.

#### Forecasts combination

The predictions of the forecasting models are combined similar to the FFORMA's combination technique. The forecasts from the base models are thus summed after they are weighted according to the meta-learner's estimated probability distribution.

#### Meta-features

The same meta-features and feature selection procedure are used as per the NN-STACK approach.

## 2.5 Details of experimentation

### 2.5.1 The evaluation protocol

We evaluate all ensembles using ten-fold cross validation, where we reinitialise each method ten times and evaluate it's performance for a 10% hold-out set. This validation process is repeated for five runs, and we average the scores collected over all fifty validation sets to rule out the chance of luck from some random initialisations.

A pseudorandom number generator (PRNG) [61] is configured to produce the indices for the splitting of the data into the training and validation sets. The seed numbers used are one, two, three, four and five for each run respectively.

## 2.5.2 Research Instruments

### Hardware and software

We implemented all algorithms in Python and ran them on a 2.60GHz GHz Intel Core i7 PC with 16 16GB RAM and 1,365 MHz Nvidia RTX 2060 GPU with 6GB GDDR6 memory. The repository of our experiment's source code is available on GitHub <sup>3</sup>.

### Data

The M4 data set is accessible on the Makridakis Open Forecasting Center's website <sup>4</sup> and consists of six subsets for each of the six frequencies to forecast. Each of these contains univariate time-series provided as numeric vectors without the conventional time stamp dependent variable. In addition, a data set is provided that contains features describing the domain and forecasting horizon for each of the reference sets.

For the weak-learner forecasts, we reuse the base models' point forecasts from the M4 competition's submissions repository <sup>5</sup>. From the M4 competition's data, we only use the training set to extract the meta-features for some ensembles that utilise them. For each run of our cross-validation, we use a different 90% splitting of our M4's test set as our ensembles' training target series and the other 10% as the holdout test set.

### Third-party libraries

We modified the Python implementation of the FFORMA <sup>6</sup> by Federico Garza and others that implements a Microsoft's LightGBM meta-learner. LightGBM has shown to produce state of the art performance against other boosting methods [62]. Our deep learning meta-learners (NN-STACK and NN-AVG) were implemented using Tensorflow's Keras module <sup>7</sup>.

We use the *tsfeatures* <sup>8</sup> Python library to extract the same time-series features as per the original FFORMA paper. Table B.1 (see Appendix B) provides a brief description of these features.

The tuning of our gradient boosting method hyperparameters are achieved using Bayesian optimisation with the Gaussian process method from the *Scikit-optimize* Python library <sup>9</sup> with minimisation configured over the OWA error.

<sup>3</sup><https://github.com/Pieter-Cawood/FFORMA-ESRNN>

<sup>4</sup><https://mofc.unic.ac.cy/the-dataset>

<sup>5</sup><https://github.com/Mcompetitions/M4-methods/tree/master/Point%20Forecasts>

<sup>6</sup><https://github.com/FedericoGarza/fforma>

<sup>7</sup>[https://www.tensorflow.org/api\\_docs/python/tf/keras](https://www.tensorflow.org/api_docs/python/tf/keras)

<sup>8</sup><https://pypi.org/project/tsfeatures>

<sup>9</sup><https://scikit-optimize.github.io>

### Preprocessing of the ensembles' input data

Only the stacking ensemble learned from the point forecasts of the base models directly. We experimented with some preprocessing of the generated forecast data; however, the model's predictions were more accurate when fit with the raw forecast values.

All ensembles, except for the simple model averaging, utilise time-series statistics that are extracted from the reference series. No additional processing was required on this data since it was already in the form of statistical measurements.

### The ensembles' architectures, hyperparameters and training

The ensembles' architectures and hyperparameters were found using the training set of the first fold of our first cross-validation run. The two ensembles that use neural networks (NN-STACK and NN-AVG) were tuned using backtesting and the gradient boosting methods (FFORMA and FFORMS) were tuned using Bayesian optimisation. The parameter search space for the Bayesian optimisation was limited based on some initial results, and the Gaussian process method was configured to estimate the parameters that minimise the OWA error over 300 runs of parameter observations.

The **FFORMA** and **FFORMS** architecture and hyperparameters were identical and they were automatically tuned using Bayesian optimisation over 300 runs of a limited parameter search space. Table 2.2 presents the hyperparameter settings used to train the FFORMA ensembles for the M4 competition's data set. Subset names H, W, D, M, Y and Q correspond to hourly, weekly, daily, monthly, yearly and quarterly data subsets.

For the model's training, an early stopping procedure was implemented that monitors the loss of the validation set that is split as a quarter of the training data set. Once the loss in the validation has not improved for ten runs, the training loop was terminated and the model with the lowest validation loss was retained.

TABLE 2.2: The FFORMA's hyperparameters.

Hyperparameter	H	W	D	M	Y	Q
n-estimators	2000	2000	2000	1200	1200	2000
min data in leaf	63	50	200	100	100	50
number of leaves	135	19	94	110	110	94
eta	0.607	0.46	0.90	0.20	0.10	0.75
max depth	61	17	9	28	28	43
subsample	0.49	0.49	0.52	0.50	0.50	0.81
colsample bytree	0.90	0.90	0.49	0.50	0.50	0.49
early stop rounds	10	10	10	10	10	10



The **NN-STACK**'s architectures and hyperparameters were determined using backtesting on the training data set of the first fold of the first run of cross-validation. The MLP architectures for the Hourly and Weekly subsets were deep with eleven hidden layers with neuron numbers of 100, 100, 100, 100, 100, 50, 50, 50, 50, 20 and 20 sequentially. A more shallow network was used to model the other subsets with three ten-neuron hidden layers.

Early stopping was configured to terminate the training loop after a few runs where no improvement was observed in the losses. Table 2.3 presents the hyperparameter settings used to train the NN-STACK ensembles for the M4 competition's data set. Subset names H, W, D, M, Y and Q correspond to hourly, weekly, daily, monthly, yearly and quarterly data subsets.

TABLE 2.3: The NN-STACK hyperparameters.

Hyperparameter	H	W	D	M	Y	Q
hidden layers	11	11	11	3	3	3
learning rate	0.0001	0.0001	0.0003	0.0003	0.0003	0.0003
batch size	1200	225	225	225	225	225
epochs	600	300	300	300	300	300
early stop rounds	15	15	15	15	15	15

Similar to the stacking method, the **NN-AVG**'s architecture and hyperparameters were determined using backtesting on the training data set of the first fold of the first run of cross-validation. The MLP architecture was deep, with eleven hidden layers of neurons in numbers of 100, 100, 100, 100, 100, 50, 50, 50, 50, 20 and 20 sequentially.

Early stopping was configured to terminate the training loop after a few runs where there was no improvement observed in the losses. Table 2.4 presents the hyperparameter settings used to train the NN-AVG ensembles for the M4 competition's data set. Subset names H, W, D, M, Y and Q correspond to hourly, weekly, daily, monthly, yearly and quarterly data subsets.

TABLE 2.4: The NN-AVG hyperparameters.

Hyperparameter	H	W	D	M	Y	Q
hidden layers	11	11	11	11	11	11
learning rate	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
batch size	1200	225	52	52	52	52
epochs	600	300	300	300	300	300
early stop rounds	15	3	3	3	3	3

### 2.5.3 Results data and analysis

We do forecasting over different horizons as per the competition requirements. We compute and record the sMAPE, MASE and OWA errors for the



test set, that is split using ten-Fold cross-validation. We use the average OWA errors as a key instrument of our analysis of the results.

We additionally summarise all of the forecasting methods' OWA errors using violin plots to help visualise their performance in terms of the range of their error distributions. Hintze and Nelson [63] proposed violin plot as an extension to the traditional box plot, that integrates a smoothed histogram from local density estimates to display the data in a single useful plot for analysis.

## 2.6 Limitations

This research is limited to experimentation with a handful of ensemble techniques showed promise to deliver state of the art forecasting performance.

The performance of the ensembles are not analysed in terms of their speed as we are only concerned regarding their accuracy as per the M4 competition requirements.

We only discuss the model architectures and hyperparameters of the ensembles implemented in this paper, as the forecasts of the base models are retrieved from a secondary source and their parameters are irrelevant for this report.

## 2.7 Ethical Considerations

This research does not require ethical clearance as it does not contain personal data and it is freely available to the public.

## Chapter 3

# Results and Discussion

This section reports the forecasting results of each method for the six data subsets of the M4 forecasting competition. Using the evaluation protocol discussed in section 2.5.1, we report all ensembles' results over fifty runs of independent data sets. We first present the average OWA errors for each method, followed by violin plots to visualise the distribution of the OWA errors and to support our discussion of the results.

### 3.1 Detailed results

Table 3.1 present key empirical results for our quantitative analysis. The table presents the average OWA performance for each forecasting method for each of the six data subsets of the M4 forecasting competition. Figures 3.1 - 3.6 depict the model OWA distributions and visualises the data quartiles and extreme values. We use this information to analyse the consistency of each forecast method's accuracy. The violin plots' upper extreme values were clipped to 3.0, which is a high OWA measurement that represents model failure.

TABLE 3.1: Average OWA performance over fifty test-folds of the M4 data set.

	H (0.4K)	W (0.4K)	D (4.2K)	M (48K)	Y (23K)	Q (24K)	Average (100K)
<b>Base models</b>							
ARIMA	0.577	0.925	1.047	0.903	0.892	0.898	0.874
Comb	1.556	0.944	<b>0.981</b>	0.920	0.867	0.890	1.026
Damped	0.936	0.992	0.999	0.924	0.890	0.893	0.939
ES-RNN	0.440	0.864	1.046	0.836	0.778	0.847	0.802
Theta	1.006	0.965	0.998	0.907	0.872	0.917	0.944
<b>Ensembles</b>							
FFORMA	<b>0.415</b>	<b>0.725</b>	0.983	<b>0.800</b>	<b>0.732</b>	<b>0.816</b>	<b>0.745</b>
FFORMS	<b>0.427</b>	<b>0.740</b>	0.984	<b>0.810</b>	<b>0.745</b>	<b>0.826</b>	<b>0.755</b>
AVG	0.847	0.860	0.985	0.863	0.804	0.856	0.869
NN-AVG	0.453	<b>0.740</b>	0.986	0.825	0.759	0.839	0.767
NN-STACK	1.427	0.810	<b>0.927</b>	0.833	0.771	0.838	0.934

Table 3.1’s column names H, W, D, M, Y and Q correspond to the Hourly, Weekly, Daily, Monthly, Yearly and Quarterly data subsets. Each method’s average OWA performance (over all data subsets) is given in the last column. The number of time-series of each data subset are provided in parenthesis and the best results are given in bold and red highlights the second-best results.

FIGURE 3.1: OWA error distributions for the Hourly subset

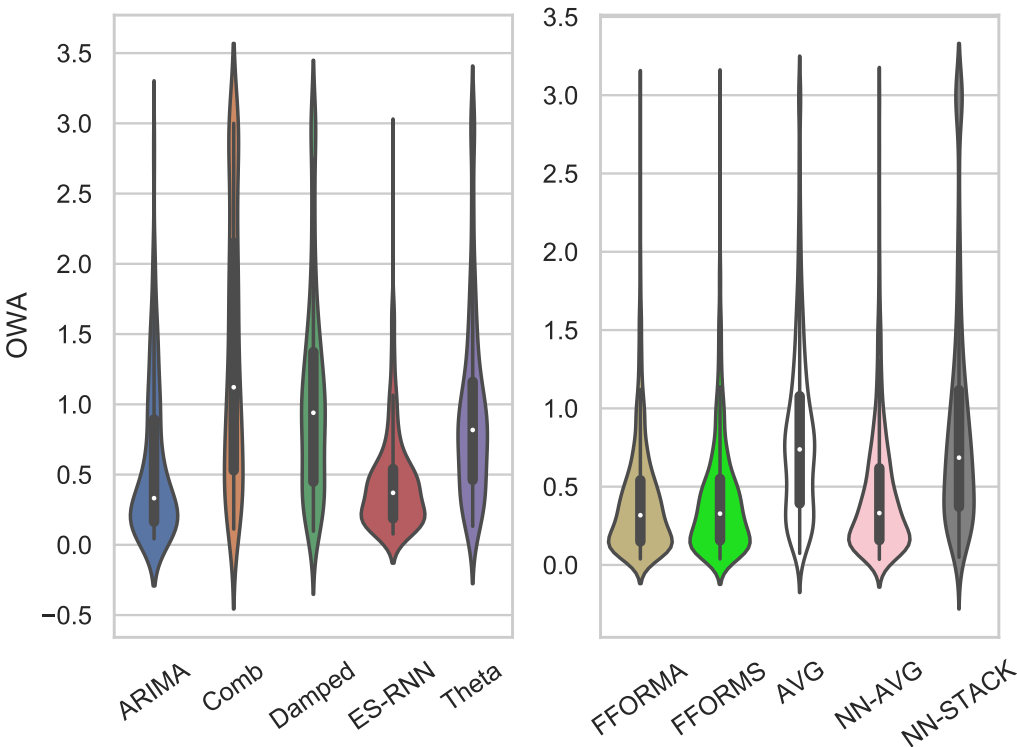
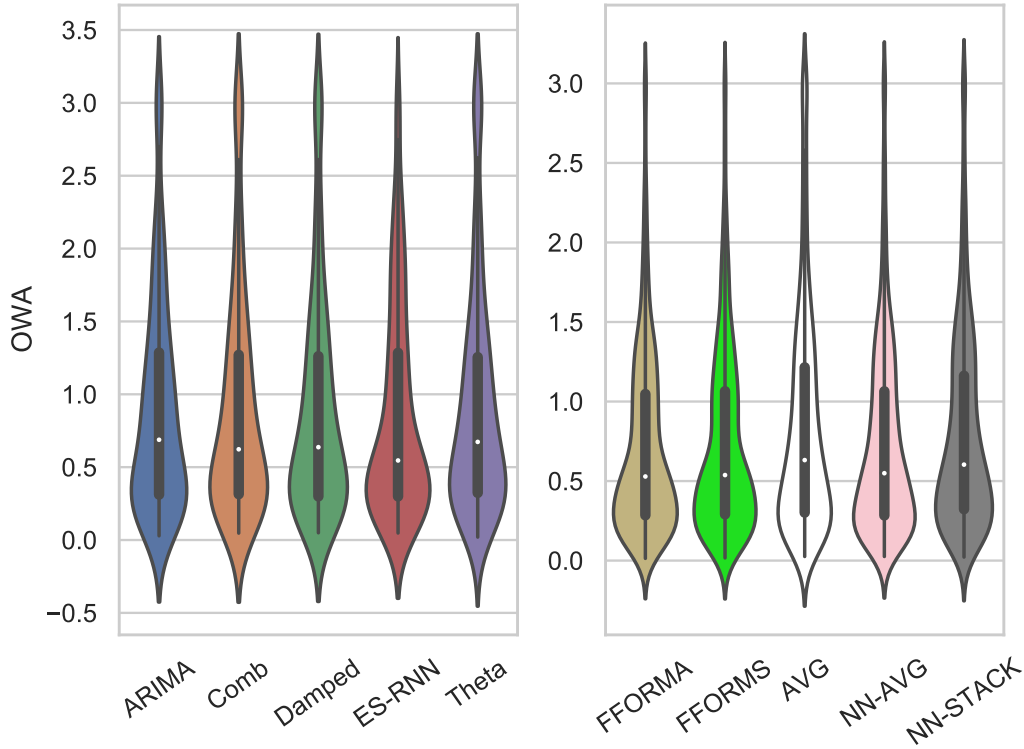


FIGURE 3.2: OWA error distributions for the Weekly subset



### 3.1.1 Observations of the Hourly subset

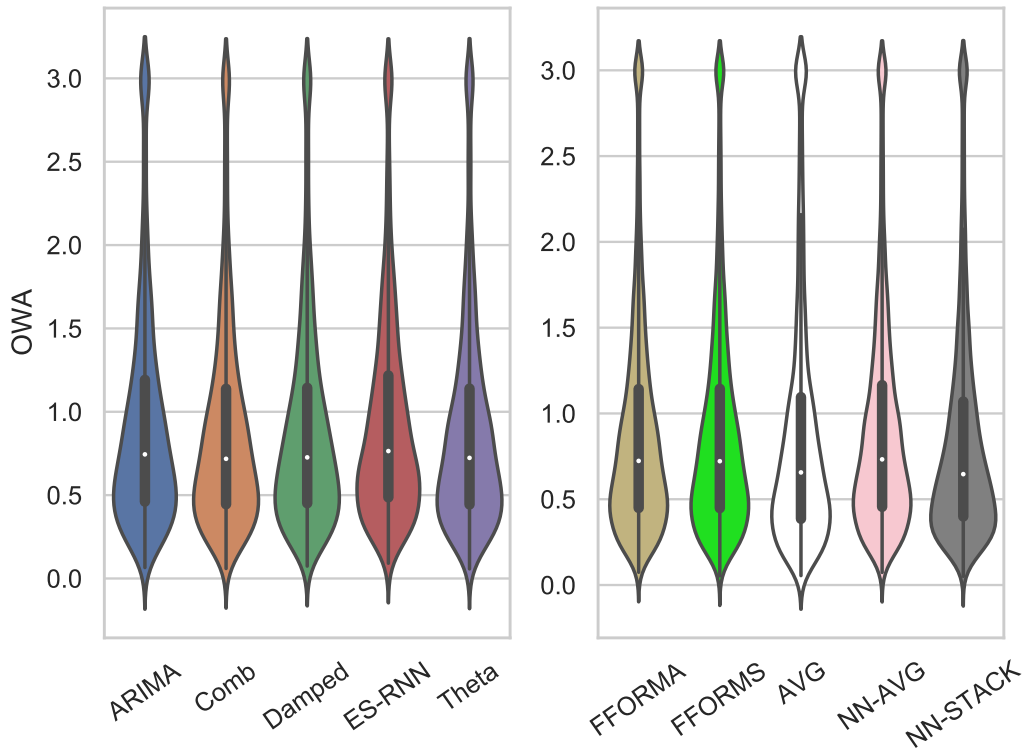
We note that for the Hourly subset, the ES-RNN produced the only stable forecasts amongst the pool of base models (see Figure 3.1.) Subsequently, the NN-STACK failed as an ensembling method and produced the experiment's worst average ensemble result. This failure was also partly due to the small data set of only 414 series and the largest forecasting horizon requirement of 48 points.

The FFORMA outperformed all other forecasting methods and produced results at least twice as good as the AVG benchmark. We note that the ES-RNN's average OWA error here might slightly be improved ( $0.440 - 0.415 = 0.025$ ), from utilising the hybrid model with the FFORMA ensemble technique.

### 3.1.2 Observations of the Weekly subset

Considering the OWA distributions depicted in Figure 3.2, we note that the ES-RNN produced the lowest average OWA error amongst the other base models, with considerably fewer forecasts reaching the extreme value of 3.0. This distinctive performance of at least one base model allowed all ensemble methods to outperform all of the base models (see Table 3.1.)

FIGURE 3.3: OWA error distributions for the Daily subset



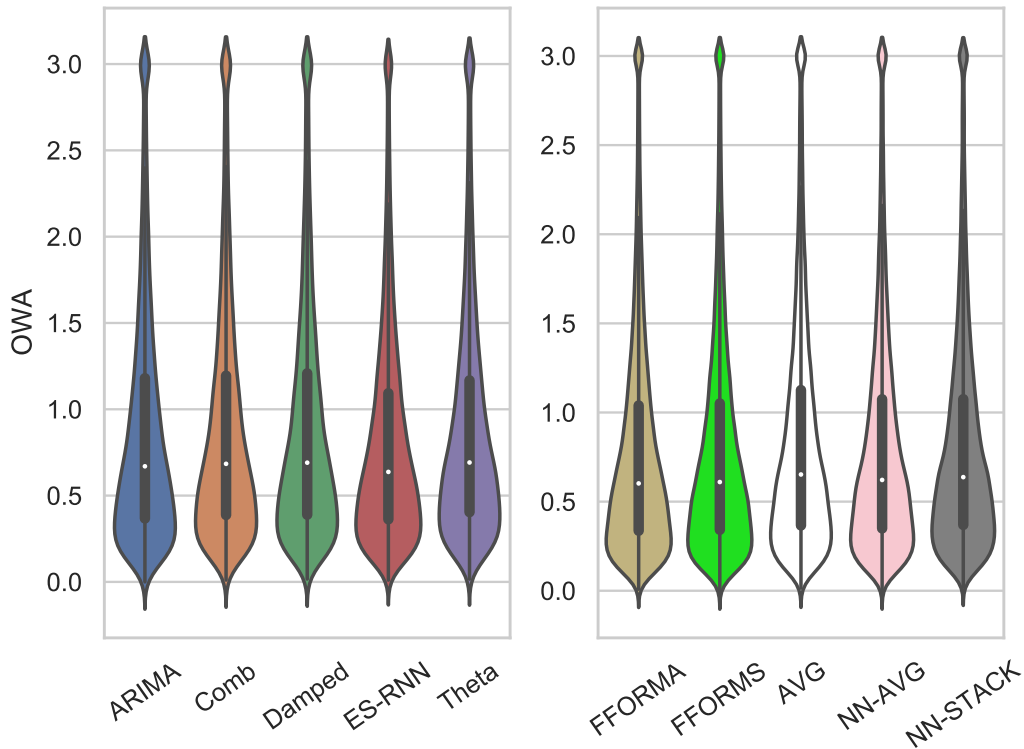
We note that the FFORMA ensemble was able to improve the ES-RNN's OWA error by a large margin ( $0.864 - 0.725 = 0.139$ ). The FFORMA performed the best of all methods, whereas the NN-AVG was the second best method and most successful neural network approach.

### 3.1.3 Observations of the Daily subset

We note from the distributions of the base model errors (see Figure 3.3), the pool of base models performed exceptionally similar. Moreover, this was the only data set where the ES-RNN failed to outperform some of the other base models. Subsequently, the gradient boosting methods failed to gain the advantage of the ES-RNN's forecasts as per the other subsets and the NN-STACK method was the only successful ensemble and best approach.

The Daily subset contains the most imbalance of time-series amounts from the different domains. Subsequently, the gradient boosting methods failed to perform since they do not utilise information regarding the time-series domains. This was also the only experiment where we observed significantly similar performance between the base models and the results suggest that the NN-STACK method is a more suitable ensemble for situations where it is difficult for the ensemble to learn the weightings of superiority amongst the pool of base models.

FIGURE 3.4: OWA error distributions for the Monthly subset



### 3.1.4 Observations of the Monthly subset

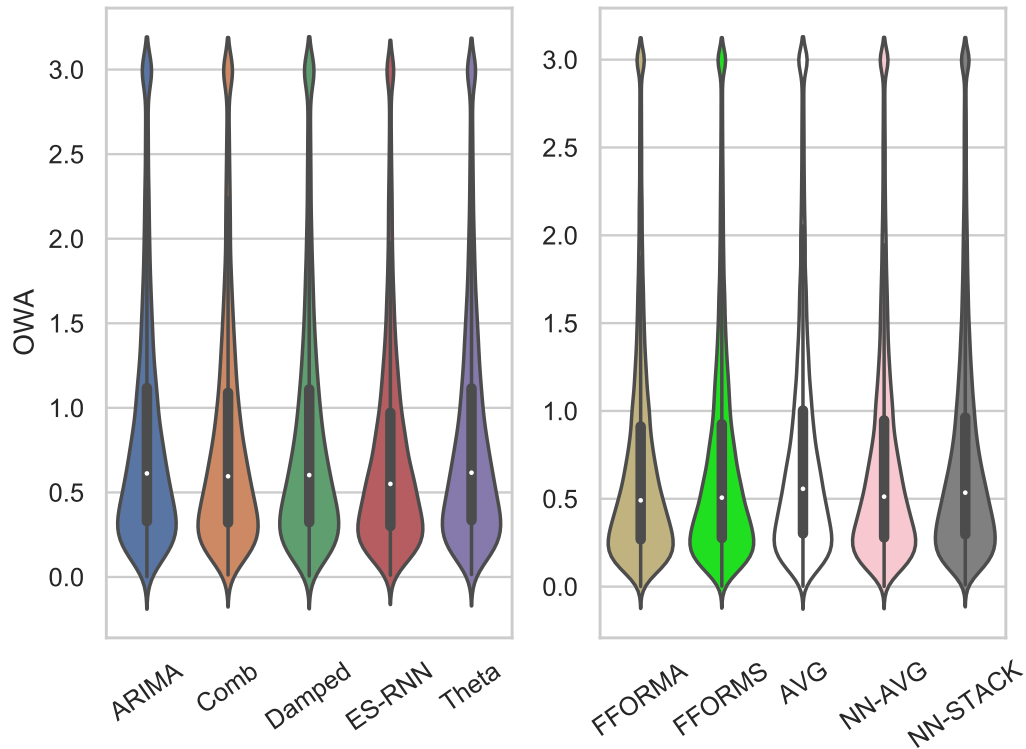
The Monthly subset consists of 48,000 time-series and it was the largest and most domain-balanced data set of all the experiments. Figure 3.4 depicts the distribution of the OWA errors, and we note that the ES-RNN performed best amongst the base models, with a smaller number of model failures. These conditions are ideal to analyse the general performance of the ensemble methods, and the results show similar performance in average OWA errors and only a slight ( $0.836 - 0.800 = 0.036$ ) improvement when the FFORMA learns from the predictions of the ES-RNN. Nevertheless, considering the average OWA error, the FFORMA remains the superior method and outperformed all other approaches.

### 3.1.5 Observations of the Yearly subset

The Yearly subset was one of the larger and nonseasonal data sets used in our experiments.

We note that all ensembles, except for the AVG benchmark, were successful at outperforming the ES-RNN slightly. We also observed similar OWA distributions between the gradient boosting and neural network ensembles (see Figure 3.5.) However, the FFORMA showed the highest improvement ( $0.778 - 0.732 = 0.046$ ) and it is the preferred ensemble for time-series with a Yearly seasonality.

FIGURE 3.5: OWA error distributions for the Yearly subset



### 3.1.6 Observations of the Quarterly subset

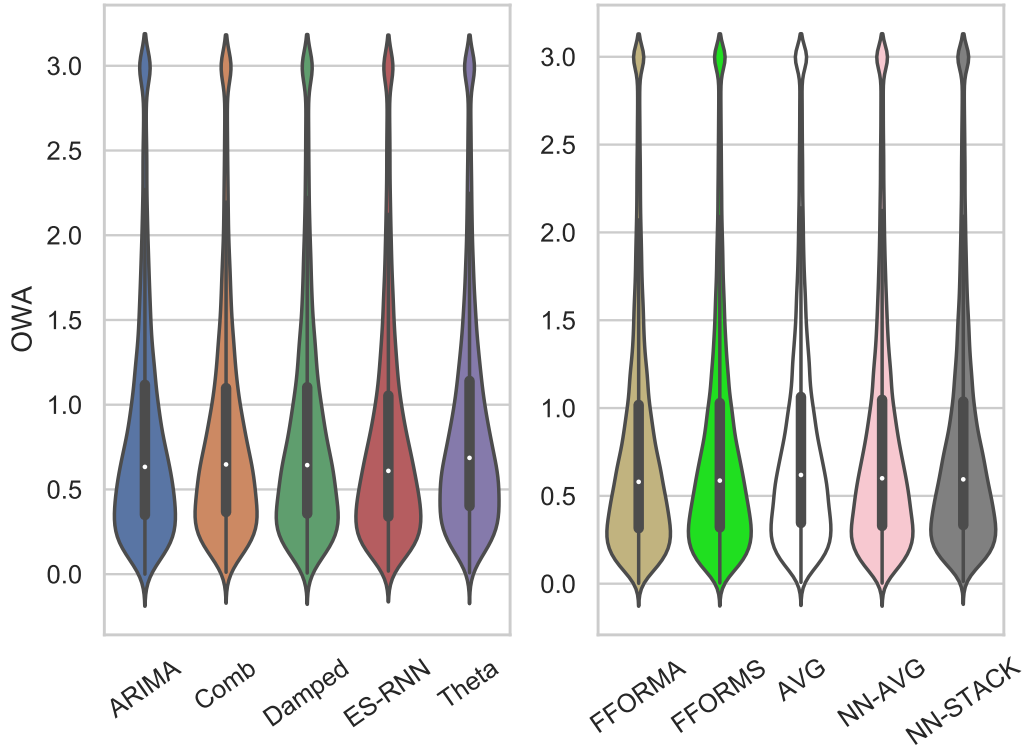
The quarterly subset is similar to that of the yearly one in terms of size, balance and forecast horizon. The only exception is that the quarterly time-series contain a higher order of seasonality.

Considering the ensembles' OWA distributions in Figure 3.6, we observed similar performance to that of the performance on the Yearly subset. Likewise, there was no base model distinguishable in terms of the OWA distributions. However, the larger data set allowed the ensemble methods to avoid the case of the smaller Daily data set (with similar performing base models), and the FFORMA ensemble produced a slight improvement ( $0.847 - 0.816 = 0.031$ ) over the average accuracy of the ES-RNN.

### 3.1.7 Observations of overall results

We added the AVG method as a benchmark for the other ensembles so that we could analyse their performance relative to something intuitive. The average OWA gap between the AVG and other ensemble methods (e.g., between the FFORMA:  $0.869 - 0.745 = 0.124$ ) indicate that machine learning is a robust solution to late data fusion.

FIGURE 3.6: OWA error distributions for the Quarterly subset



Considering the OWA distributions (Figures 3.4 - 3.6) of the three larger subsets (Monthly, Yearly and Quarterly), we find that although all of the ensembles beat the ES-RNN's average error, there is still generally a deal of uncertainty. Nevertheless, the ensembles were still effective to lower the upper quantiles of the OWA distributions by a notable margin. A larger improvement was observed in the OWA distributions of the three smaller subsets (see Figures 3.1 - 3.3), where the weak learners showed more distinctive performance (considering their average OWA errors.) This provides evidence that i) ensemble learning's performance is dependent on distinctive performance of its weak learners and ii) ensemble learning's performance is dependent on the diversity of the data (i.e., the similarity of the time-series from the different domains.)

We note that the gradient boosting ensembles generally outperform those of neural networks. The average OWA gap between the top two methods of each (i.e., the FFORMA and NN-AVG:  $0.767 - 0.745 = 0.022$ ) is enough for us to confirm the superiority of gradient boosting as an ensemble method for time-series forecasting. Furthermore, both of the gradient boosting ensembles outperforms all other methods on all subsets except for the daily time-series. Table 3.1 statistically proves that for all data sets, it is always possible to improve a standalone time-series forecast model by adding it to an ensemble with other statistical forecast models. Lastly, we note that since we utilise the ES-RNN as a base model of the FFORMA, we were able to have



the FFORMA outperform the ES-RNN (the M4 winner) for all six subsets of data.

## Chapter 4

# Conclusion

Our primary objective of this research was to compare numerous ensemble methods against the state of the art forecasting method, the ES-RNN, to determine whether we might i) improve the accuracy of the ES-RNN and ii) single out any method that we empirically determined is the state of the art forecasting technique for future research efforts.

We reviewed time-series forecasting research and we noted that there was a gap in the literature of research that determines the boost of accuracy one might achieve from implementing late data fusion with the ES-RNN. Moreover, there was a need of research that compares the performance of different types of ensembles using the same forecasts from a pool of forecasting models. Consequently, our research questions were aligned with answering the extent of which the forecast accuracy of the ES-RNN might be improved from ensemble learning.

We undertook confirmatory experimental research, using the M4 forecasting competition's data set, and to promote unbiased results, our experiment used a ten-fold cross-validation procedure and we averaged all results over five independent runs. We realised that averaging the results might be misleading and we additionally analysed the results using plots of the forecast methods' OWA error distributions. The experiment include four statistical base models, namely ARIMA, Comb, Damped and Theta. And an advanced machine learning and statistical hybrid model, the ES-RNN. The base models are used with different ensemble strategies of model averaging, model stacking and model selection. The ensembles of the experiment include two gradient boosting ones, namely the FFORMA and FFORMS; two neural network approaches, namely NN-STACK and NN-AVG; and a naive arithmetic average as a benchmark.

In respect of the obtained results, we infer that the FFORMA is a state of the art ensemble technique that might be used to boost the performance of the ES-RNN. However, considering the case of the Daily subset, the NN-STACK approach might be a more suitable tool when the pool of forecasting models have similar performance. None of the ensembles were able to prevent the occurrence of model failure (where the measured OWA was greater than 3.0); however, we observed that the upper quartiles of the ensembles' OWA distributions were notably lower than those of the base models. Further, we find weighted model averaging to be a superior ensemble approach as both the FFORMA and NN-AVG outperformed versions of their same architecture that do model stacking or selection.

The stacking approach performs the task of regression over single points of the base model forecasts. This removes the temporal element of the data and the meta-learner instead learns a generalised combination function. We tried stacking the models using a recurrent neural network and the results were poor. However, we remain firm in our belief that a stacking approach conscious of the data's temporality should outperform model averaging, and we suggest future work consider this.

## Appendix A

# The FFORMA algorithm

---

**Algorithm 1:** FFORMA's forecast combination [9].

---

### OFFLINE PHASE: TRAINING

#### Inputs

- $\{x_1, x_2, \dots, x_N\}$  : N observed time series from the reference set.
- $\{f_1, f_2, \dots, f_F\}$  : a set of F functions to calculate the meta-features.
- $\{m_1, m_2, \dots, m_M\}$  : a set of M forecasting models.

#### Output

Meta-learner model

#### Prepare the metadata

**for**  $n \leftarrow 1$  **to**  $N$  **do**

1. Split  $x_n$  to a training and test series using time-splitting.
2. Calculate the meta-features  $f_n \in F$  over the training period.
3. Fit each base forecasting method  $m \in M$  and generate forecasts.
4. Compute the forecast losses  $L_{nm}$  over the test period.

**end**

#### Train the meta-learner

Train the meta-learner model using the meta-data and forecasting errors of the base models, by minimising:

$$\arg \min_w \sum_{n=1}^N \sum_{m=1}^M w(f_n)_m L_{nm}$$

### ONLINE PHASE: FORECASTING

#### Inputs

The meta-learner model from the offline phase.

- $\{x_1, x_2, \dots, x_N\}$  : N observed time series from the validation set.
- $\{m_1, m_2, \dots, m_M\}$  : the set of M forecasting models.

#### Output

$\{y_1, y_2, \dots, y_N\}$  : Forecast for each series in the test set.

**for**  $n \leftarrow 1$  **to**  $N$  **do**

1. Calculate the meta-features  $f_n \in F$ .
  2. Use the meta-learner to produce  $w(f_n)$ , an M-vector of weights.
  3. Generate the forecasts for each of the  $M$  forecasting methods.
  4. Combine each of the forecasts using  $w$  to produce the final forecast.
- 3-4. FFORMS: Select from  $M$  the model with the highest allocated  $w$  to produce the final forecast.

**end**

\*FFORMS modification that replace the FFORMA forecast steps 3 & 4.

---

## Appendix B

# The FFORMA meta-features

TABLE B.1: Features used in the FFORMA framework [9].

	Feature	Description	Nonseasonal	Seasonal
1	T	length of time series	Yes	Yes
2	trend	strength of trend	Yes	Yes
3	seasonality	strength of seasonality	-	Yes
4	linearity	linearity	Yes	Yes
5	curvature	curvature	Yes	Yes
6	spikiness	spikiness	Yes	Yes
7	e_acf1	first ACF value of remainder series	Yes	Yes
8	e_acf10	sum of squares of first 10 ACF values of remainder series	Yes	Yes
9	stability	stability	Yes	Yes
10	lumpiness	lumpiness	Yes	Yes
11	entropy spectral	entropy	Yes	Yes
12	hurst	Hurst exponent	Yes	Yes
13	nonlinearity	nonlinearity	Yes	Yes
14	alpha	$ETS(A, A, N)\hat{\alpha}$	Yes	Yes
15	beta	$ETS(A, A, N)\hat{\beta}$	Yes	Yes
16	hwalpha	$ETS(A, A, A)\hat{\alpha}$	-	Yes
17	hwbeta	$ETS(A, A, A)\hat{\beta}$	-	Yes
18	hwgamma	$ETS(A, A, A)\hat{\gamma}$	-	Yes
19	ur_pp	test statistic based on Phillips–Perron test	Yes	Yes
20	ur_kpss	test statistic based on KPSS test	Yes	Yes
21	y_acf1	first ACF value of the original series	Yes	Yes
22	diff1y_acf1	first ACF value of the differenced series	Yes	Yes
23	diff2y_acf1	first ACF value of the twice-differenced series	Yes	Yes
24	y_acf10	sum of squares of first 10 ACF values of original series	Yes	Yes
25	diff1y_acf10	sum of squares of first 10 ACF values of differenced series	Yes	Yes
26	diff2y_acf10	sum of squares of first 10 ACF values of twice-differenced series	Yes	Yes
27	seas_acf1	autocorrelation coefficient at first seasonal lag	-	Yes
28	sediff_acf1	first ACF value of seasonally differenced series	-	Yes
29	y_pacf5	sum of squares of first 5 PACF values of original series	Yes	Yes
30	diff1y_pacf5	sum of squares of first 5 PACF values of differenced series	Yes	Yes
31	diff2y_pacf5	sum of squares of first 5 PACF values of twice-differenced series	Yes	Yes
32	seas_pacf	partial autocorrelation coefficient at first seasonal lag	Yes	Yes
33	crossing_point	number of times the time series crosses the median	Yes	Yes
34	flat_spots	number of flat spots from 10 equidistant points	Yes	Yes
35	nperiods	number of seasonal periods in the series	-	Yes
36	seasonal_period	length of seasonal period	-	Yes
37	peak	strength of peak	Yes	Yes
38	trough	strength of trough	Yes	Yes
39	ARCH.LM	ARCH LM statistic	Yes	Yes
40	arch_acf	sum of squares of the first 12 autocorrelations of z2	Yes	Yes
41	garch_acf	sum of squares of the first 12 autocorrelations of r2	Yes	Yes
42	arch_r2	R2 value of an AR model applied to z2	Yes	Yes
43	garch_r2	R2 value of an AR model applied to r2	Yes	Yes

# Bibliography

- [1] S. Makridakis, R. J. Hyndman, and F. Petropoulos, "Forecasting in social settings: The state of the art," *International Journal of Forecasting*, vol. 36, no. 1, pp. 15–28, 2020.
- [2] J. Burton, "Robert fitzroy and the early history of the meteorological office," *The British Journal for the History of Science*, pp. 147–176, 1986.
- [3] G. Udny Yule, "On a method of investigating periodicities in disturbed series, with special reference to wolfer's sunspot numbers," *Philosophical Transactions of the Royal Society of London Series A*, vol. 226, pp. 267–298, 1927.
- [4] J. G. De Gooijer and R. J. Hyndman, "25 years of time series forecasting," *International journal of forecasting*, vol. 22, no. 3, pp. 443–473, 2006.
- [5] G. Zhang, B. E. Patuwo, and M. Y. Hu, "Forecasting with artificial neural networks:: The state of the art," *International journal of forecasting*, vol. 14, no. 1, pp. 35–62, 1998.
- [6] N. G. Reich, L. C. Brooks, S. J. Fox, *et al.*, "A collaborative multiyear, multimodel assessment of seasonal influenza forecasting in the united states," *Proceedings of the National Academy of Sciences*, vol. 116, no. 8, pp. 3146–3154, 2019.
- [7] C. J. McGowan, M. Biggerstaff, M. Johansson, *et al.*, "Collaborative efforts to forecast seasonal influenza in the united states, 2015–2016," *Scientific reports*, vol. 9, no. 1, pp. 1–13, 2019.
- [8] M. A. Johansson, K. M. Apfeldorf, S. Dobson, *et al.*, "An open challenge to advance probabilistic forecasting for dengue epidemics," *Proceedings of the National Academy of Sciences*, vol. 116, no. 48, pp. 24 268–24 274, 2019.
- [9] P. Montero-Manso, G. Athanasopoulos, R. J. Hyndman, and T. S. Talagala, "Fforma: Feature-based forecast model averaging," *International Journal of Forecasting*, vol. 36, no. 1, pp. 86–92, 2020.
- [10] S. Smyl, "A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting," *International Journal of Forecasting*, vol. 36, no. 1, pp. 75–85, 2020, M4 Competition.
- [11] L. Rokach, "Ensemble-based classifiers," *Artificial intelligence review*, vol. 33, no. 1, pp. 1–39, 2010.
- [12] Q. Wu, "A hybrid-forecasting model based on gaussian support vector machine and chaotic particle swarm optimization," *Expert Systems with Applications*, vol. 37, no. 3, pp. 2388–2394, 2010.

- [13] M. Khashei and M. Bijari, "A new class of hybrid models for time series forecasting," *Expert Systems with Applications*, vol. 39, no. 4, pp. 4344–4357, 2012.
- [14] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "Statistical and machine learning forecasting methods: Concerns and ways forward," *PLoS ONE*, vol. 13, Mar. 2018.
- [15] B. N. Oreshkin, D. Carпов, N. Chapados, and Y. Bengio, "N-beats: Neural basis expansion analysis for interpretable time series forecasting," *arXiv preprint arXiv:1905.10437*, 2019.
- [16] G. R. Shinde, A. B. Kalamkar, P. N. Mahalle, N. Dey, J. Chaki, and A. E. Hassanien, "Forecasting models for coronavirus disease (covid-19): A survey of the state-of-the-art," *SN Computer Science*, vol. 1, no. 4, pp. 1–15, 2020.
- [17] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "The m4 competition: 100,000 time series and 61 forecasting methods," *International Journal of Forecasting*, vol. 36, no. 1, pp. 54–74, 2020.
- [18] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE transactions on evolutionary computation*, vol. 1, no. 1, pp. 67–82, 1997.
- [19] B. Arinze, "Selecting appropriate forecasting models using rule induction," *Omega*, vol. 22, no. 6, pp. 647–658, 1994.
- [20] A. E. Raftery, D. Madigan, and J. A. Hoeting, "Bayesian model averaging for linear regression models," *Journal of the American Statistical Association*, vol. 92, no. 437, pp. 179–191, 1997.
- [21] B. Clarke, "Comparing bayes model averaging and stacking when model approximation error cannot be ignored," *Journal of Machine Learning Research*, vol. 4, no. Oct, pp. 683–712, 2003.
- [22] M. H. D. M. Ribeiro and L. dos Santos Coelho, "Ensemble approach based on bagging, boosting and stacking for short-term prediction in agribusiness time series," *Applied Soft Computing*, vol. 86, p. 105837, 2020.
- [23] P. Cawood and T. L. van Zyl, "Feature-weighted stacking for nonseasonal time series forecasts: A case study of the covid-19 epidemic curves," in *2021 8th International Conference on Soft Computing Machine Intelligence (ISCMI)*, 2021, pp. 53–59.
- [24] A. C. Lorena, A. I. Maciel, P. B. de Miranda, I. G. Costa, and R. B. Prudêncio, "Data complexity meta-features for regression problems," *Machine Learning*, vol. 107, no. 1, pp. 209–246, 2018.
- [25] S. Barak, M. Nasiri, and M. Rostamzadeh, "Time series model selection with a meta-learning approach; evidence from a pool of forecasting algorithms," *arXiv preprint arXiv:1908.08489*, 2019.
- [26] G. P. Zhang, "Time series forecasting using a hybrid arima and neural network model," *Neurocomputing*, vol. 50, pp. 159–175, 2003.

- [27] N. Liu, Q. Tang, J. Zhang, W. Fan, and J. Liu, "A hybrid forecasting model with parameter optimization for short-term load forecasting of micro-grids," *Applied Energy*, vol. 129, pp. 336–345, 2014.
- [28] J.-Z. Wang, Y. Wang, and P. Jiang, "The study and application of a novel hybrid forecasting model—a case study of wind speed forecasting in china," *Applied Energy*, vol. 143, pp. 472–488, 2015.
- [29] Y. Qin, K. Li, Z. Liang, *et al.*, "Hybrid forecasting model based on long short term memory network and deep learning neural network for wind signal," *Applied energy*, vol. 236, pp. 262–272, 2019.
- [30] A. Aksoy, N. Öztürk, and E. Sucky, "Demand forecasting for apparel manufacturers by using neuro-fuzzy techniques," *Journal of Modelling in Management*, 2014.
- [31] W. Deng, G. Wang, and X. Zhang, "A novel hybrid water quality time series prediction method based on cloud model and fuzzy forecasting," *Chemometrics and Intelligent Laboratory Systems*, vol. 149, pp. 39–49, 2015.
- [32] R. Rahmani, R. Yusof, M. Seyedmahmoudian, and S. Mekhilef, "Hybrid technique of ant colony and particle swarm optimization for short term wind energy forecasting," *Journal of Wind Engineering and Industrial Aerodynamics*, vol. 123, pp. 163–170, 2013.
- [33] S. Kumar, S. K. Pal, and R. Singh, "A novel hybrid model based on particle swarm optimisation and extreme learning machine for short-term temperature prediction using ambient sensors," *Sustainable Cities and Society*, vol. 49, p. 101 601, 2019.
- [34] G. S. Atsalakis and K. P. Valavanis, "Surveying stock market forecasting techniques—part ii: Soft computing methods," *Expert systems with applications*, vol. 36, no. 3, pp. 5932–5941, 2009.
- [35] J. E. Hanke and D. W. Wichern, "Business forecasting," 2005.
- [36] S. Makridakis, "Accuracy measures: Theoretical and practical concerns," *International journal of forecasting*, vol. 9, no. 4, pp. 527–529, 1993.
- [37] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *International journal of forecasting*, vol. 22, no. 4, pp. 679–688, 2006.
- [38] A. Redd, K. Khin, and A. Marini, "Fast es-rnn: A gpu implementation of the es-rnn algorithm," *arXiv preprint arXiv:1907.03329*, 2019.
- [39] J. Fattah, L. Ezzine, Z. Aman, H. El Moussami, and A. Lachhab, "Forecasting of demand using arima model," *International Journal of Engineering Business Management*, vol. 10, p. 1 847 979 018 808 673, 2018.
- [40] R. J. Hyndman and Y. Khandakar, "Automatic time series forecasting: The forecast package for r," *Journal of statistical software*, vol. 27, no. 1, pp. 1–22, 2008.
- [41] V. Assimakopoulos and K. Nikolopoulos, "The theta model: A decomposition approach to forecasting," *International journal of forecasting*, vol. 16, no. 4, pp. 521–530, 2000.



- [42] E. Ostertagová and O. Ostertag, "The simple exponential smoothing model," in *The 4th International Conference on Modelling of Mechanical and Mechatronic Systems, Technical University of Košice, Slovak Republic, Proceedings of conference*, 2011, pp. 380–384.
- [43] C. C. Holt, "Forecasting seasonals and trends by exponentially weighted moving averages," *International journal of forecasting*, vol. 20, no. 1, pp. 5–10, 2004.
- [44] E. McKenzie and E. S. Gardner Jr, "Damped trend exponential smoothing: A modelling viewpoint," *International Journal of Forecasting*, vol. 26, no. 4, pp. 661–665, 2010.
- [45] S. Makridakis and M. Hibon, "The m3-competition: Results, conclusions and implications," *International journal of forecasting*, vol. 16, no. 4, pp. 451–476, 2000.
- [46] A. Nazim and A. Afthanorhan, "A comparison between single exponential smoothing (ses), double exponential smoothing (des), holt's (brown) and adaptive response rate exponential smoothing (arres) techniques in forecasting malaysia population," *Global Journal of Mathematical Analysis*, vol. 2, no. 4, pp. 276–280, 2014.
- [47] P. S. Kalekar *et al.*, "Time series forecasting using holt-winters exponential smoothing," *Kanwal Rekhi school of information Technology*, vol. 4329008, no. 13, pp. 1–13, 2004.
- [48] S. Chang, Y. Zhang, W. Han, *et al.*, "Dilated recurrent neural networks," *arXiv preprint arXiv:1710.02224*, 2017.
- [49] F. Petropoulos, R. J. Hyndman, and C. Bergmeir, "Exploring the sources of uncertainty: Why does bagging for time series forecasting work?" *European Journal of Operational Research*, vol. 268, no. 2, pp. 545–554, 2018.
- [50] F. Chan and L. L. Pauwels, "Some theoretical results on forecast combinations," *International Journal of Forecasting*, vol. 34, no. 1, pp. 64–74, 2018.
- [51] E. S. Gardner Jr, "Exponential smoothing: The state of the art," *Journal of forecasting*, vol. 4, no. 1, pp. 1–28, 1985.
- [52] R. Hyndman, A. B. Koehler, J. K. Ord, and R. D. Snyder, *Forecasting with exponential smoothing: the state space approach*. Springer Science & Business Media, 2008.
- [53] J. W. Taylor, "Short-term electricity demand forecasting using double seasonal exponential smoothing," *Journal of the Operational Research Society*, vol. 54, no. 8, pp. 799–805, 2003.
- [54] G. Claeskens, N. L. Hjort, *et al.*, "Model selection and model averaging," *Cambridge Books*, 2008.
- [55] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.

- [56] R. J. Hyndman, E. Wang, and N. Laptev, "Large-scale unusual time series detection," in *2015 IEEE international conference on data mining workshop (ICDMW)*, IEEE, 2015, pp. 1616–1619.
- [57] T. S. Talagala, R. J. Hyndman, G. Athanasopoulos, *et al.*, "Meta-learning how to forecast time series," *Monash Econometrics and Business Statistics Working Papers*, vol. 6, p. 18, 2018.
- [58] A. Liaw, M. Wiener, *et al.*, "Classification and regression by random-forest," *R news*, vol. 2, no. 3, pp. 18–22, 2002.
- [59] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
- [60] J. H. Zar, "Spearman rank correlation," *Encyclopedia of biostatistics*, vol. 7, 2005.
- [61] L. Blum, M. Blum, and M. Shub, "A simple unpredictable pseudo-random number generator," *SIAM Journal on computing*, vol. 15, no. 2, pp. 364–383, 1986.
- [62] E. Al Daoud, "Comparison between xgboost, lightgbm and catboost using a home credit dataset," *International Journal of Computer and Information Engineering*, vol. 13, no. 1, pp. 6–10, 2019.
- [63] J. L. Hintze and R. D. Nelson, "Violin plots: A box plot-density trace synergism," *The American Statistician*, vol. 52, no. 2, pp. 181–184, 1998.