Arbitrated Ensemble for Time Series Forecasting

Vítor Cerqueira[1,2], Luís Torgo[1,2], Fábio Pinto[1,2], and Carlos Soares[1,2]

[1] University of Porto, Portugal
[2] INESC TEC, Porto, Portugal
vmac@inesctec.pt, ltorgo@inesctec.pt, fhpinto@inesctec.pt, csoares@fe.up.pt

Abstract. This paper proposes an ensemble method for time series forecasting tasks. Combining different forecasting models is a common approach to tackle these problems. State-of-the-art methods track the loss of the available models and adapt their weights accordingly. Metalearning strategies such as stacking are also used in these tasks. We propose a metalearning approach for adaptively combining forecasting models that specializes them across the time series. Our assumption is that different forecasting models have different areas of expertise and a varying relative performance. Moreover, many time series show recurring structures due to factors such as seasonality. Therefore, the ability of a method to deal with changes in relative performance of models as well as recurrent changes in the data distribution can be very useful in dynamic environments. Our approach is based on an ensemble of heterogeneous forecasters, arbitrated by a metalearning model. This strategy is designed to cope with the different dynamics of time series and quickly adapt the ensemble to regime changes. We validate our proposal using time series from several real world domains. Empirical results show the competitiveness of the method in comparison to state-of-the-art approaches for combining forecasters.

Keywords: Dynamic Ensembles, Metalearning, Time Series, Numerical Prediction, Reproducible Research

1 Introduction

Time series forecasting is an important topic in the machine learning research community with several applications across different domains. Time series often comprise non-stationarities and time-evolving complex structures which difficult the forecasting process.

To cope with these issues a common approach is to combine several forecasting models. Combination strategies typically involve tracking the error of the available models and adaptively weigh them accordingly. Using stacking [30], a metalearning approach, to combine the available forecasting models is also a common approach. This method directly models inter-dependencies between models, which might be relevant to take into account the diversity among them [5].

In this paper we adopt a metalearning strategy to adaptively combine the available forecasting models. However, contrary to stacking, we model the individual expertise of each forecasting model and specialize them across the time series. Consequently, the forecasting models are combined in such a way that they are only picked for predicting in examples that they are good at. Moreover, as opposed to tracking the error on past instances, our combination approach is more proactive as it is based on predictions of future loss of models. This can result in a faster adaptation to changes in the environment.

The motivation for our approach is that different learning models have different areas of expertise across the input space [5]. In time series forecasting there is evidence that forecasting models have a varying relative performance over time [2]. Moreover, it is also common for the underlying process generating the time series to have recurrent structures due to factors such as seasonality [12]. In this context, we hypothesize that the metalearning strategy enables the ensemble to better detect changes in the relative performance of models or changes between different regimes and quickly adapt itself to the environment.

Specifically, our proposed metalearning strategy, hereby denoted as ADE (Arbitrated Dynamic Ensemble), is based on an Arbitrating architecture [21]. In that sense, we build a meta-learner for each base-learner that is part of the ensemble. Each meta-learner is specifically designed to model how apt its base counterpart is to make a prediction for a given test example. This is accomplished by analysing how the error incurred by a given learning model relates to the characteristics of the data. At test time, the base-learners are weighted according to their degree of competence in the input observation, estimated by the predictions of the meta-learners. This is illustrated in Figure 1.

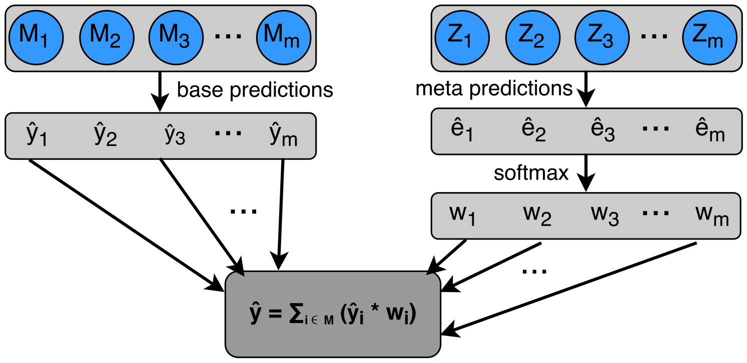

Fig. 1: Example scheme of the workflow of ADE for a new prediction. The base learners M produce the predictions $\hat{y}_i$, $i \in \{1, \ldots, m\}$ for the next value of the time series. In parallel, the meta learners Z produce the weights w of each base learner according to their predictions of error ($\hat{e}_i$). The final prediction $\hat{y}$ is computated using a weighted average of the predictions relative to the weights.

While a given base learner M_i is trained to model the future values of the time series, its metalearning associate Z_i is trained to model the error of M_i. Z_i then can make predictions regarding the error that M_i will incur when predicting the future values of the time series. In effect, the larger the estimates produced by Z_i (relative to the other models in the ensemble) the lower the weight of M_i will be in the combination rule. Although arbitrating models is not generally new [21], it was originally formulated for selecting classifiers. Moreover, we address several of its limitations which significantly improve its overall ability.

We validate the proposed method in 14 real-world time series. Empirical experiments suggest that our method is competitive against different adaptive methods for combining base-learners and other metalearning approaches such as stacking [30]. We note that all experiments are fully reproducible. Both the methods and time series data sets are publicly available as an R software package.

In summary, the contributions of this paper are:

– ADE, an arbitrated dynamic ensemble. Arbitrating was originally proposed for dynamic selection of classifiers [21], whereas we adapt it to time series forecasting tasks;
– We introduce a blocked prequential procedure in the Arbitrating approach to obtain out-of-bag predictions in the training set in order to increase the data used to train the metalearning model;
– A softmax function used to weight the forecasters. This function is commonly used in neural networks, however, to the best of our knowledge, it has not been applied yet to dynamically weight base-learners;
– An empirical study comparing different training strategies for the base and meta levels including a discussion about its implications in terms of predictive performance and computational resources used.

We start by outlining the related work in Section 2; the methodology is addressed in Section 3, where we formalise ADE and explain our contributions; the experiments and respective results are presented and discussed in Section 4, where we also include a brief scalability analysis of ADE. Finally, Section 5 concludes the paper.

2 Related Work

In this section we briefly revise the state-of-the-art methods for adaptively combining models for time series forecasting tasks, listing their characteristics and limitations as well as highlighting our contributions.

The simple average of the predictions of the available base-learners have been shown to be a robust combination method [7, 26, 20]. Nonetheless, other more sophisticated approaches have been proposed.

2.1 Windowing Strategies for Model Combination

AEC is a method for adaptively combining forecasters [25]. It uses an exponential re-weighting strategy to combine forecasters according to their past performance. It includes a forgetting factor to give more importance to recent values.

Timmermann argues that for the prediction of stock returns models have only short-lived periods of predictability [27]. He proposes an adaptive combination based on the recent R^2 of forecasters. If all models have poor explained variance (low R^2) in the recent observations then the forecast is set to the mean value of those observations. Otherwise, the base-learners are combined by averaging their predictions with the arithmetic mean.

Newbold and Granger proposed a method which weighs models in a linear way according to their performance in recent past data [19].

The outlined models are related to our work in the sense that they employ adaptive heuristics to combine forecasters. However, these heuristics are incremental or sliding summary statistics on relative past performance. Conversely, we explore differences among base-learners to specialise them across the data space. Moreover, we use a more proactive heuristic that is based on the prediction of relative future performance of individual forecasters.

2.2 Metalearning Strategies for Model Combination

Metalearning provides a way for modeling the learning process of a learning algorithm [4]. Several methods follow this approach to improve the combinination or selection of models [30, 22]. The most popular strategy to combine forecasters is similar to stacking [30]. Linear regression is used to adaptively estimate the weights of the base-learners [8].

Our proposal follows a metalearning strategy called arbitrating. This was introduced before for dynamic selection of classifiers [21] (Figure 1). A prediction is made using a combination of different classifiers that are selected according to their expertise concerning the input data. The expertise of a model is learned using a meta-model, one for each available base classifier, which models the loss of its base counterpart. At runtime, the classifier with the highest confidence is selected to make a prediction. This work follows a first attempt on adapting the arbitration approach to numerical prediction tasks. This strategy was applied to solar radiation forecasting tasks, and the arbitration methodology is extended by weighing all forecasting models, as opposed to selecting the one with lowest predicted error [6]. In this paper we introduce other components to arbitrating. These address several of its drawbacks, such as its inefficient use of the available data, by using OOB samples from the training set; a more robust combination rule by using a committee of recent well performing models and weighing them using a softmax function; and the general translation to the time series forecasting tasks, which is fundamentally different than classification tasks.

3 Arbitrating for Time Series Forecasting

A time series Y is a temporal sequence of values $Y = \{y_1, y_2, \ldots, y_t\}$, where y_i is the value of Y at time i. We use time delay embedding to represent Y in an Euclidean space with embedding dimension K. Effectively, we construct a set of observations which are based on the past K lags of the time series.

This is accomplished by mapping the time series Y into the embedding vectors $V_{N-K+1} = \{\mathbf{v_1}, \mathbf{v_2}, \ldots \mathbf{v_{N-K+1}}\}$ where each $\mathbf{v_i} = \langle y_{i-(K-1)}, y_{i-(K-2)}, \ldots, y_i \rangle$.

The proposed methodology in ADE for time series forecasting settles on the three main steps: (**i**) An offline training step of M, the set of base-learners which are used to forecast future values of Y, and the online iterative steps: (**ii**) Training or updating of meta-learners Z, which model the expertise of base-learners, and (**iii**) prediction of y_{t+1} using M, dynamically weighed according to Z.

3.1 Learning Base-level Models

In the offline learning phase we train m individual forecasters. Each $M^j, \forall j \in \{1, ..., m\}$ is built using the available time series Y_{tr}^K. The objective is to predict the next value of the series, y_{t+1}. We use the subscripts tr and ts to denote the training and testing sets, respectively.

By embedding the time series to an Euclidean space we are able to apply standard regression learning models. In this context, M is comprised by a set of heterogeneous models, such as Gaussian Processes and Neural Networks. Heterogeneous models have different inductive biases and assumptions regarding the process generating the data. Effectively, we expect models to have different expertise across the time series. The learning step is described in Algorithm 1.

3.2 Learning Meta-level Models

In the metalearning step of ADE the goal is to build algorithms capable of modelling the expertise of each model across the data-space. Our working hypothesis is that the ensemble can leverage individual learners with different inductive biases to better cope with the different regimes causing the time series.

Our assumption is that not all models will perform equally well at any given prediction point. This idea is in accordance with findings reported in prior work [2]. Systematic evidence was found that some models have varying relative performance over time and that other models are persistently good (or bad) throughout the time series. Furthermore, in many environments the dynamic concepts have a recurring nature, due to, for instance, seasonality. In effect, we use the meta learners to dynamically weigh base learners and adapt the combined model to changes in the relative performance of the base models, as well as for the presence of different regimes in the time series.

Algorithm 1 Learning M

```
Require: Available Time Series $Y$; Embedding Dimension $K$
1: Embed $Y$ into $Y_{tr}^K$
2: for all $M^j \in M$ do
3:     train $M^j$ using $Y_{tr}^K$
4: Return $M$
```

Our metalearning approach is based on an arbitrating architecture originally introduced in [21] (c.f. Section 2 for an explanation of arbitration). Specifically, a meta-learner $Z^j, \forall j \in \{1, \ldots, m\}$ is trained to build the following model:

$$e^j = f(I) \tag{1}$$

where e^j is the absolute error incurred by M^j, I is the metafeature set and f is the regression function. The metafeatures are the embedding vectors, i.e., the past values of the time series.

We perform this regression analysis on a meta-level to understand how the error of a given model relates to the dynamics and the structure of the time series. Effectively, we can capitalise on this knowledge by dynamically combining base-learners according to the expectation of how they will perform.

Blocked Prequential for Out-of-Bag Predictions In the original formulation of the Arbitrating strategy, the metalearning layer only starts at run-time, using only information from test observations [21]. This is motivated by the need for unbiased samples to build reliable metalearners. However, this means that at the beginning, few observations are available to train the meta learners, which might result in underfitting.

ADE uses the training set to produce out-of-bag (OOB) predictions which are then used to compute an unbiased estimate of the loss of each base learner. By retrieving OOB samples from the training set we are able to significantly increase the amount of data available to the meta learners. We hypothesize that this strategy improves the overall performance of the ensemble by improving the accuracy of each metalearner.

We produce OOB samples by running a blocked prequential procedure [9]. The available embedded time series used for training Y_{tr}^K is split into β equally-sized and sequential blocks of contiguous observations. In the first iteration, the first block is used to train the base learners M and the second is used to test them. Then, the second block is merged with the first one for training M and the third block is used for testing. This procedure continues until all blocks are tested. In summary, each metalearner uses all the information available up to the prediction point. However, each metalearner is trained every λ observations, so less computation is used to update the models.

Committee of Models for Prediction As described earlier, the predictive performance of forecasting models has been reported to vary over a given time series. We address this issue with a committee of models, rather than selecting a single model [21], where we trim recently poor performing models from the combination rule for an upcoming prediction (e.g. [14]). Formally, we maintain the $\alpha\%$ base learners with lowest mean absolute error in the last Ω observations, trimming off the remaining ones. The predictions of the meta-level models are used to weigh the selected forecasters.

Additionally, if a base learner is consistently out of the committee, its meta-level pair is not trained. This saves computational resources. The meta-learning phase is described in Algorithm 2.

Algorithm 2 Metalearning Z

```
Require: Available observations at runtime $Y_{ts}$; Embedding Dimension $K$; Training
    Signal $\lambda$; meta-committee ${}^{\alpha}Z$
1: Embed $Y_{ts}$ into $Y_{ts}^{K} \rightarrow I$
2: for all $Z^j \in {}^{\alpha}Z$ and not trained in the last training signal $\lambda$ do
3:     train $Z^j$ to model: $e^j = f(I)$
4: Return ${}^{\alpha}Z$
```

3.3 Predicting y_{t+1}

For a new observation y_{t+1}, ADE combines the base learners in M according to the meta information obtained from the models in Z to generate a prediction.

Initially we form the ${}^{\alpha}M$ committee with the $\alpha\%$ models in M with best performance in the last Ω observations. The corresponding metalearners are also discarded from the upcoming prediction so we also form the meta-committee ${}^{\alpha}Z$.

The weigh of a base learner M^j in ${}^{\alpha}M$ is given by the softmax of its negative predicted loss. This is formalised by the following equation:

$$w_{t+1}^{j} = \frac{exp(-\hat{e}^j)}{\sum_{j \in {}^{\alpha}M} exp(-\hat{e}^j)} \tag{2}$$

where $\hat{e}_{t+1}^{j}$ is the prediction made by ${}^{\alpha}Z^j$ for the absolute loss that ${}^{\alpha}M^j$ will incur in y_{t+1}, w_{t+1}^{j} is the weigh of M_j for observation y_{t+1} and exp denotes the exponential function. With the application of the softmax function, which is widely used in the modelling process of neural networks, the weigh of a given model decays exponentially as its predicted loss increases. We use this function (instead of a more traditional linear transformation) to further increase the influence of the predicted best performing models.

The final prediction is a weighted average of the predictions made by the base-learners $\hat{y}^j$ with respect to w_{t+1}^{j}: $\hat{y}_{t+1} = \sum_{j \in {}^{\alpha}M} \hat{y}_{t+1}^{j} \cdot w_{t+1}^{j}$.
The prediction step of ADE is described in Algorithm 3.

4 Experiments

In this section we present the experiments carried out to validate ADE. These address the following research questions:

Q1: How does the performance of the proposed method relates to the performance of the state-of-the-art methods for time series forecasting tasks and state-of-the-art methods for combining forecasting models?

Q2: Is it beneficial to use out-of-bag predictions from the training set to increase the data used to train the meta learners?
Q3: Is it beneficial to use a weighing scheme in our Arbitrating strategy instead of selecting the predicted best forecaster as originally proposed [21]?
Q4: How does the performance of ADE vary by the introduction of a committee, where poor recent base-learners are discarded from the upcoming prediction, as opposed to weighing all the models?
Q5: Does a non-linear weighting strategy using a softmax function produce better estimates instead of a traditional linear scaling?
Q6: How does the performance of ADE vary by using different updating strategies for the base and meta models?

To address these questions we used 14 real world time series from four different domains, briefly described in Table 1. In the interest of computational efficiency we truncated the time series to 2000 values.

Algorithm 3 Predicting y_{t+1}

```
Require: $K$, $M$, $Z$, $\alpha$, $\Omega$, ${}^{\alpha}M$, ${}^{\alpha}Z$
1: Embed the previous $K-1$ values into $Y^{K}_{t+1}$
2: Get meta-predictions $\hat{e}^{j}_{t+1}$ from $Z^{j} \in {}^{\alpha}Z$
3: Compute weights $w^{j}_{t+1} = exp(-\hat{e}^{j}_{t+1}) / \sum_{Z^{j} \in {}^{\alpha}Z} exp(-\hat{e}^{j}_{t+1})$
4: Get predictions $\hat{y}^{j}_{t+1}$ from models $M^{j} \in {}^{\alpha}M$
5: Compute final prediction $\hat{y}_{t+1} = \sum_{j=1}^{m} \hat{y}^{j}_{t+1} \cdot w^{j}_{t+1}$
6: Add $y_{t+1}$ to $Y_{ts}$
7: Update Committees ${}^{\alpha}M$ and ${}^{\alpha}Z$ according to $\alpha$ and $\Omega$
8: Return $\hat{y}_{t+1}$ and go back to the metalearning step (Algorithm 2)
```

4.1 Experimental Setup

The methods used in the experiments were evaluated using the mean squared error (MSE) on 10 Monte Carlo repetitions. For each repetition, a random point in time is chosen from the full time window available for each series, and the previous window N consisting of 50% of the data set size is used for training the ensemble while the following window of size 30% is used for testing. The results obtained by the different methods were compared using the non-parametric Wilcoxon Signed Rank test. The experiments were carried out using performanceEstimation [28] R package.

We tested two different embedding dimensions by setting K to **7** and **15**.

4.2 Ensemble Setup and Baselines

The base-models M comprising the ensemble are the following:

Table 1: Datasets and respective summary

ID	Time series	Data source	Data characteristics
1	Electricity Total Load	Hospital Energy Loads [10]	Hourly values from Jan. 1, 2016 to Mar. 25, 2016
2	Equipment Load		
3	Water Heating Load		
4	Gas Energy		
5	Gas Heat Energy		
6	Ameal	Oporto Water Demand from different locations [1]	Half-hourly values from Feb. 6, 2013 to Mar. 19, 2013
7	Preciosa Mar		
8	Montes Burgos		
9	Global Horiz. Radiation	Solar Radiation Monitoring [3]	Hourly values from Feb. 16, 2016 to May 5, 2016
10	Direct Normal Radiation		
11	Diffuse Horiz. Radiation		
12	Sea Level Pressure	Ozone Level Detection [17]	Half-hourly from Feb. 6, 2013 to Mar. 13, 2013
13	Geo-Potential Height		
14	K index		

SVM Support Vector Machines [15];
NN Feed Forward Neural Nets [29];
GP Gaussian Processes [15];
GLM Generalized Linear Models [11];
RF Random Forests [31];
GBR Generalized Boosted Regr. [24];
MARS MARS [18];
RBR Rule-based Regression [16];
PPR Projection Pursuit Regr. [23].

Different parameter settings are used for each of the individual learners, adding up to **40** models.

We use a Random Forest as meta-learner. The parameter λ was set to **10**, which means that at run-time the metalearners are re-trained every 10 observations. Each set contains the OOB samples from the training set and the observations from the test set up to the upcoming prediction. In the blocked prequential procedure used to obtain OOB samples we used 10 folds, i.e., β equal to **10**. The committee for each prediction contains the 50% forecasters with best performance in the last 50 observations (α and Ω values are set to **50**). We drop only half the models in the interest of keeping the combined model readily adaptable to changes in the environment. An average performing model may suddenly become important and the combined model should be able to capture these situations. By setting Ω to 50 we strive for estimates of recent performance that renders a robust committee. We compare the performance of ADE against the following 8 approaches:

Stacking: An adaptation of stacking [30] for times series, where a meta-model is learned using the base-level predictions as attributes. To preserve the temporal order of observations, the out-of-bag predictions used to train the metalearner (a random forest) are obtained using a blocked prequential procedure (c.f. Section 3.2). We tried different strategies for training the metalearner (e.g. holdout) but blocked prequential presented the best results;

Arbitrating: The original arbitrating approach [21], c.f. Section 2;

S: A static heterogeneous ensemble. All base learners are simply averaged using the arithmetic mean [26];
S-W: A weighted linear combination of the models, with weights according to their performance using all past information [14];
AEC: The adaptive combination procedure AEC [25], c.f. Section 2;
S-WRoll: The adaptive combination method, with a linear combination of the forecasters according to their recent performance [19];
ERP: The adaptive combination procedure proposed by Timmermann [27], c.f. Section 2;
ARIMA: A state-of-the-art method for time series forecasting. We use the implementation in the forecast R package [13], which automatically tunes ARIMA to an optimal parameter setting;

The following variants of ADE were tested:

ADE-Arb: A variant of ADE in which at each time point the best model is selected to make a prediction. Here best is the one with lowest predicted loss. This is in accordance with the original arbitrating architecture [21];
ADE-meta-runtime: A variant of ADE in which there is no blocked prequential procedure to obtain OOB samples to increase the data provided to the metalearners. In this scenario Z is trained in data obtained only at run-time, which is also in accordance with the original arbitrating strategy;
ADE-all-models: A variant of ADE, but without the formation of a committee. In this case, all forecasting models are weighed according to their expertise in the input data;
ADE-linear-committee: A variant of ADE, but using a linear transformation for weighting the base learners according to their predicted loss, instead of the proposed softmax;
ADE-meta-GP: A variant of ADE, but using a Gaussian Process with a linear kernel as meta-learner instead of a Random Forest;

All the variants of the proposed method use a random forest as meta-learner.

4.3 Results

Table 2 presents the paired comparisons between the proposed method, ADE, and the baselines. The numbers represent wins and losses of the proposed method. The numbers in parenthesis represent statistically significant wins and losses. The average rank for each model is also presented with the corresponding deviation. Figures 2 and 3 represent the critical difference diagrams for the post-hoc Bonferroni-Dunn test relative to the other baselines in the literature and baselines which are variants of ADE, respectively. In the interest of space we present the post-hoc test results only for K equal to 7. The analysis for K equal to 15 leads to similar conclusions.

The proposed method achieves competitive performance with respect to the baselines and ADE variants. Relative to **S**, **S-W**, **AEC**, **S-WRoll** and **ERP**, state of the art approaches for combining individual forecasters, and **ARIMA**,

Table 2: Paired comparisons between the proposed method and the baselines for different embedding dimensions in the 14 time series. The Rank column stands for the average rank and respective standard deviation of each model. A rank of 1 in an experiment means that the model was the best method.

Method	K = 7			K = 15		
	ADE			ADE		
	Looses	Wins	Rank	Looses	Wins	Rank
S	3 (2)	**11** (11)	10.0 ± 3.3	3 (2)	**11** (10)	9.9 ± 3.3
S-W	5 (4)	**9** (8)	6.3 ± 3.0	6 (3)	**8** (8)	6.6 ± 3.8
ARIMA	3 (3)	**11** (11)	11.1 ± 5.5	3 (3)	**11** (11)	10.6 ± 5.3
Stacking	7 (5)	7 (5)	5.9 ± 5.2	3 (1)	**11** (8)	7.0 ± 4.2
Arbitrating	4 (0)	**10** (6)	7.8 ± 4.8	1 (0)	**13** (9)	9.0 ± 3.9
AEC	4 (2)	**10** (10)	8.6 ± 3.9	2 (0)	**12** (11)	9.3 ± 2.7
S-WRoll	5 (4)	**9** (7)	7.4 ± 3.4	4 (2)	**10** (9)	8.1 ± 3.6
ERP	3 (2)	**11** (11)	10.0 ± 3.3	3 (2)	**11** (11)	10.6 ± 3.8
ADE-Arb	1 (0)	**13** (7)	8.4 ± 3.1	2 (0)	**12** (8)	6.9 ± 3.9
ADE-meta-runt.	6 (0)	**8** (5)	5.8 ± 3.9	3 (1)	**11** (6)	5.6 ± 3.0
ADE-all-models	5 (2)	**9** (3)	6.0 ± 2.3	3 (2)	**11** (3)	5.1 ± 3.3
ADE-linear-com.	5 (4)	**9** (6)	6.4 ± 3.3	5 (5)	**9** (7)	6.1 ± 3.2
ADE-meta-GP	6 (0)	**8** (6)	6.1 ± 3.0	4 (3)	**10** (7)	6.0 ± 2.6
ADE	–	–	$\mathbf{5.2 \pm 2.8}$	–	–	$\mathbf{4.1 \pm 3.1}$

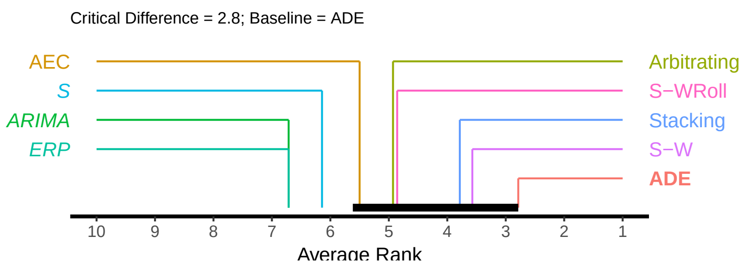

Fig. 2: Critical difference diagram for the post-hoc Bonferroni-Dunn test relative to baselines ($K = 7$)

a state of the art method for time series forecasting, the difference is significant. These results answer the research question **Q1** regarding the performance of ADE relative to the state-of-the-art approaches for time series forecasting tasks.

Comparing the proposed method against **Stacking**, a widely used metalearning strategy for combining models, the performance is no significantly different, although our method presents a better average rank. Relative to the original arbitrating architecture, denoted as **Arbitrating**, the proposed method shows a systematic improvement, which results in a much better average rank. This proves that the introduced components are fundamental for the achieved performance, which answers question **Q3** regarding the comparison between ADE and the original arbitration approach. In the following, we discuss the effect of different components in the results.

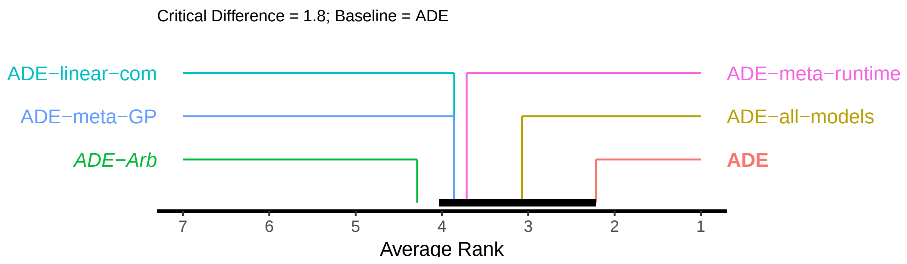

Fig. 3: Critical difference diagram for the post-hoc Bonferroni-Dunn test relative to ADE variants ($K = 7$)

ADE also shows consistent advantage over the performance of **ADE-meta-runtime** and **ADE-all-models**. This suggests that indeed it is worthwhile to get OOB predictions from the available data to improve the fit of the metalearners and to prune the ensemble for each prediction (as opposed to combining all the forecasters). These results answer our research questions **Q2** and **Q4** regarding the comparison of ADE to the respective components. The comparison between ADE and **ADE-linear-committee** shows that the softmax function renders a superior predictive performance relative to the linear transformation. In effect, a non-linear transformation indeed produces better estimates than a traditional linear scaling (question **Q5**). Comparing the proposed method against **ADE-Arb**, which used the predicted best model for forecasting, the difference is statistically significant, according to the post-hoc Bonferroni-Dunn test.

Regarding the embedding dimension, different values do not seem to alter the results significantly. Conversely, using Random Forests in the metalearning layer produced better results than with Gaussian Processes.

4.4 Further Analysis

Meta-level Performance We analysed the performance of the metalearning layer. The evaluation measure is mean absolute error. That is, the average absolute difference between the predicted error and the actual error that the base-level models incurred.

Figure 4 presents the average rank and respective deviation of each meta-model in Z, across the datasets, grouped by type of base-level learner. Here, we focus on the Random Forests as meta-level algorithm. The numbers suggest that the RBR models are the most predictable, while the NN are the least predictable ones.

Analyzing Training Strategies In this section we address the research question **Q6**. In a dynamic environment it is common to update the model over time, either online or in chunks of observations. This is because time-dependent data is prone to changes in the underlying distribution and continuous training of models ensures that one has an up-to-date model. Since ADE settles on two

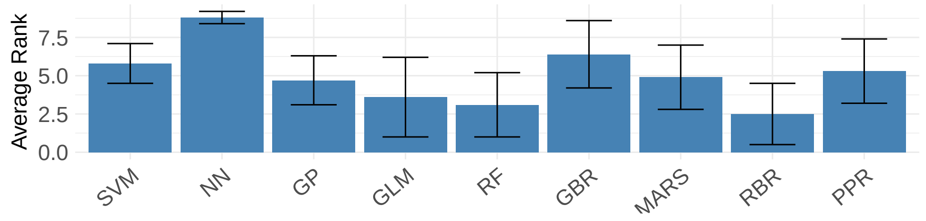

Fig. 4: Mean rank of the meta-models in terms of accuracy for predicting the performance of models obtained with several algorithms across the 14 datasets ($K = 7$) and corresponding standard-deviation.

layers of models we analysed different approaches for updating these and study their implications in predictive performance and computational resources used.

In the main experiments the base learners are not updated and meta-learners are updated in chunks of λ observations. Besides this strategy (which we denote as **A**) we try three other approaches:

B: M and Z are both trained only in the training set;
C: Both M and Z are re-trained every λ observations, which is particularly interesting if the models in M are typical online methods (e.g. ARIMA);
D: M is re-trained every λ observations but Z is trained only in the training data.

In these experiments we set λ to 1 meaning that re-training occurs at every prediction step. The test size is set to 15% of the time series total length. Other parameters follow the setup in the main experiments. We compare the four above-mentioned strategies in the 14 problems using the mean absolute scaled error (MASE). This is a scale invariant metric and is defined as follows:

$$\frac{\sum_{i=1}^{n} |e_i|}{\frac{n}{n-1}\sum_{i=2}^{n} |y_i - y_{i-1}|} \tag{3}$$

The results of this analysis are presented in Table 3. These show the average MASE and average runtime and corresponding deviations of ADE using the different retraining strategies across the 14 problems. The results for the predictive performance are comparable across the methods, although the ones that update the metalearners show a slightly lower error.

In terms of computational effort the strategies that update the base learners are significantly more expensive. Retraining the meta learners is faster than updating the base learners because of the formation of the committee in which only half the meta learners (according to the experimental setup) are retrained.

Finally, the results suggest that updating the meta learners and not updating the base learners (A) is better than the inverted strategy (D), both in predictive performance and runtime. In particular, the difference in computational time is mainly due to the selection of models included in the committee. In strategy A,

A	B	C	D
0.60±0.18	0.62±0.17	0.60±0.18	0.62±0.17

(a)

A	B	C	D
22.4±6.6	2.1±0.5	156.4±38.2	130.3±31.5

(b)

Table 3: Average MASE and respective deviation of the different retraining strategies in terms of predictive performance (a) and computational time spent in minutes (b) across the 14 problems.

the models outside the committee are not updated since they will not be used. Conversely, in strategy D, every base learner is updated.

4.5 Discussion

We empirically showed the advantages of the proposed method with respect to several state-of-the-art approaches for time series forecasting tasks.

One of the main limitations of ADE is that it is not able to directly model inter-dependencies between forecasters, which might be important to account for the diversity among models. We plan to address this issue in the future. Nonetheless, its performance is competitive with the widely used stacking method [30], which directly models these dependencies.

Some of the design decisions behind ADE are based on prior work regarding the variance in relative performance of forecasting models over a time series [2] and with potential recurring structures with the time series. However, there are cases in which time series change into new concepts and base learners may get outdated. Although we do not explicitly cover these scenarios, a simple strategy to address this issue is to track the loss of the ensemble. If its performance decreases beyond tolerance new base-learners are introduced (e.g. [12]) or existing ones are re-trained. Since an arbitration approach provides a modular architecture, models can be added (or removed) as needed.

5 Conclusions

We presented a new adaptive ensemble method for time series forecasting tasks. Our strategy for adaptively combining forecasters is based on metalearning, which provides a way of learning about the learning process of models [4]. Consequently, we are able to model their expertise in the different parts of the data and adapt the combined model to changes in the underlying environment and/or changes in relative performance of base learners. By analyzing each forecaster separately we specialize them in the sense that they will not be used for prediction – or they do but with a minor relevance – in observations that they are bad at.

ADE is motivated by the assumption that different forecasting models have varying performance over time. This is justified by prior evidence in that direction [2]. Moreover, it is not uncommon for time series to show recurrent structures.

We proved the competitiveness of our approach in fourteen real-world time series against several baselines. These include state-of-the-art methods for adaptively combining forecasters and the most widely used metalearning strategy for model combination (stacking [30]). The effect of the different adaptations of arbitrating on the results was empirically analised.

We argue that despite the competitive predictive performance, the proposed method does not directly model inter-dependencies between the available learners. We plan to investigate this issue in future work.

In the interest of reproducible science that the authors support, all methods and datasets are publicly available as an *R* software package[3].

Acknowledgements

This work is financed by the ERDF - European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme within project POCI-01-0145-FEDER-006961, and by National Funds through the FCT - Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) as part of project UID/EEA/50014/2013; Project "NORTE-01-0145-FEDER-000036" is financed by the North Portugal Regional Operational Programme (NORTE 2020), under the PORTUGAL 2020 Partnership Agreement, and through the European Regional Development Fund (ERDF). This work was partly funded by the ECSEL Joint Undertaking, the framework programme for research and innovation horizon 2020 (20142020) under grant agreement number 662189-MANTIS-2014-1.

References

1. ADDP: Oporto water consumption. `http://addp.pt`, accessed: 2016-11-21
2. Aiolfi, M., Timmermann, A.: Persistence in forecasting performance and conditional combination strategies. Journal of Econometrics 135(1), 31–53 (2006)
3. Andreas, A..W., ;, S.: Solar Radiation Monitoring Station (SoRMS): Humboldt State University, Arcata. California; NREL Rep DA-5500-56515 (2007)
4. Brazdil, P., Carrier, C.G., Soares, C., Vilalta, R.: Metalearning: Applications to data mining. Springer Science & Business Media (2008)
5. Brown, G., Kuncheva, L.I.: "good" and "bad" diversity in majority vote ensembles. In: Proceeding of Multiple Classifier Systems (MCS 2010). pp. 1–15 (2010)
6. Cerqueira, V., Torgo, L., Soares, C.: Arbitrated ensemble for solar radiation forecasting. In: International Work-Conference on Artificial Neural Networks. pp. 720–732. Springer, Cham (2017)
7. Clemen, R.T., Winkler, R.L.: Combining economic forecasts. Journal of Business & Economic Statistics 4(1), 39–46 (1986)

[3] tsensembler: `https://github.com/vcerqueira/tsensembler`

8. Crane, D.B., Crotty, J.R.: A two-stage forecasting model: Exponential smoothing and multiple regression. Management Science 13(8), B–501 (1967)
9. Dawid, A.P.: Present position and potential developments: Some personal views: Statistical theory: The prequential approach. Journal of the Royal Statistical Society. Series A (General) pp. 278–292 (1984)
10. EERE: Commercial and residential hourly load profiles tmy3 loc. in the usa. `http://en.openei.org/datasets/files/961/pub/`, accessed: 2016-11-21
11. Friedman, J., Hastie, T., Tibshirani, R.: Regularization paths for generalized linear models via coordinate descent. Journal of Statistical Software 33(1), 1–22 (2010)
12. Gama, J., Kosina, P.: Tracking recurring concepts with meta-learners. In: Portuguese Conference on Artificial Intelligence. pp. 423–434. Springer (2009)
13. Hyndman, R.J., with contributions from George Athanasopoulos, Razbash, S., Schmidt, D., Zhou, Z., Khan, Y., Bergmeir, C., Wang, E.: forecast: Forecasting functions for time series and linear models (2014), R package version 5.6
14. Jose, V.R.R., Winkler, R.L.: Simple robust averages of forecasts: Some empirical results. International Journal of Forecasting 24(1), 163–169 (2008)
15. Karatzoglou, A., Smola, A., Hornik, K., Zeileis, A.: kernlab – an S4 package for kernel methods in R. Journal of Statistical Software 11(9), 1–20 (2004)
16. Kuhn, M., Weston, S., Keefer, C., code for Cubist by Ross Quinlan, N.C.C.: Cubist: Rule- and Instance-Based Regression Modeling (2014), R package version 0.0.18
17. Lichman, M.: UCI machine learning repository (2013), `archive.ics.uci.edu/ml`
18. Milborrow, S.: earth: Multivariate Adaptive Regression Spline Models. Derived from mda:mars by Trevor Hastie and Rob Tibshirani. (2012)
19. Newbold, P., Granger, C.W.: Experience with forecasting univariate time series and the combination of forecasts. Journal of the Royal Statistical Society. Series A (General) pp. 131–165 (1974)
20. Oliveira, M., Torgo, L.: Ensembles for time series forecasting. In: ACML Proceedings of Asian Conference on Machine Learning. JMLR: Workshop and Conference Proceedings (2014)
21. Ortega, J., Koppel, M., Argamon, S.: Arbitrating among competing classifiers using learned referees. Knowledge and Information Systems 3(4), 470–490 (2001)
22. Pinto, F., Soares, C., Mendes-Moreira, J.: Chade: Metalearning with classifier chains for dynamic combination of classifiers. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Springer (2016)
23. R Core Team: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria (2013)
24. Ridgeway, G.: gbm: Generalized Boosted Regression Models (2015), R package version 2.1.1
25. Sánchez, I.: Adaptive combination of forecasts with application to wind energy. International Journal of Forecasting 24(4), 679–693 (2008)
26. Timmermann, A.: Forecast combinations. Handbook of economic forecasting 1, 135–196 (2006)
27. Timmermann, A.: Elusive return predictability. International Journal of Forecasting 24(1), 1–18 (2008)
28. Torgo, L.: An Infra-Structure for Performance Estimation and Experimental Comparison of Predictive Models (2013), R package version 0.1.1
29. Venables, W.N., Ripley, B.D.: Modern Applied Statistics with S. Springer, New York, fourth edn. (2002), iSBN 0-387-95457-0
30. Wolpert, D.H.: Stacked generalization. Neural networks 5(2), 241–259 (1992)
31. Wright, M.N.: ranger: A Fast Implementation of Random Forests (2015), R package