A Drift-Based Dynamic Ensemble Members Selection Using Clustering for Time Series Forecasting

Amal Saadallah(✉), Florian Priebe, and Katharina Morik

Artificial Intelligence Group, Department of Computer Science, TU Dortmund,
Dortmund, Germany
{amal.saadallah,florian.priebe,katharina.morik}@tu-dortmund.de

Abstract. Both complex and evolving nature of time series structure make forecasting among one of the most important and challenging tasks in time series analysis. Typical methods for forecasting are designed to model time-evolving dependencies between data observations. However, it is generally accepted that none of them is universally valid for every application. Therefore, methods for learning heterogeneous ensembles by combining a diverse set of forecasts together appear as a promising solution to tackle this task. Hitherto, in classical ML literature, ensemble techniques such as stacking, cascading and voting are mostly restricted to operate in a static manner. To deal with changes in the relative performance of models as well as changes in the data distribution, we propose a drift-aware meta-learning approach for adaptively selecting and combining forecasting models. Our assumption is that different forecasting models have different areas of expertise and a varying relative performance. Our method ensures dynamic selection of initial ensemble base models candidates through a performance drift detection mechanism. Since diversity is a fundamental component in ensemble methods, we propose a second stage selection with clustering that is computed after each drift detection. Predictions of final selected models are combined into a single prediction. An exhaustive empirical testing of the method was performed, evaluating both generalization error and scalability of the approach 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: Model clustering · Dynamic ensemble · Meta-learning · Drift-detection

1 Introduction

Time series forecasting is one of the most challenging tasks in time series analysis due to the dynamic behavior of this data type, which may involve non-stationary and complex processes [6,18,28]. Forecasting has considerably attracted the

U. Brefeld et al. (Eds.): ECML PKDD 2019, LNAI 11906, pp. 678–694, 2020.
https://doi.org/10.1007/978-3-030-46150-8_40

attention of both academic and industrial communities and has always been one of the principal steps in real-time decision-making and planning across various applications such as traffic prediction, weather forecasts, stock market prices prediction [18].

Several machine learning methods have been successfully applied to time series forecasting either by dealing with the data as an ordered sequence of observation in an offline or a streaming fashion, or by using an embedding of the time series to reformulate the forecasting task as a regression task [6]. However, it is generally accepted that none of the ML methods is universally valid for every task, in particular for forecasting. Therefore, one reasonable solution is to combine the opinion of a diverse set of models using an ensemble method. Ensembles consist of a collection of several models (i.e., experts) that are combined together to address the same defined task [22]. Ensemble construction can be divided into three main stages: (i) base model generation, where n multiple possible hypotheses are formulated to model a given phenomenon; (ii) model pruning, where only a subset of $m < n$ hypotheses is kept and (iii) model integration, where these hypotheses are combined together in one single model.

Most of the existing methods for ensemble learning on time series are focusing on optimizing the last stage [6,22,28]. Combination strategies can be grouped into three main families [30]. The first family relies on voting approaches using majority or (weighted) average votes to decide for the final output (e.g. bagging [3]). The second main family englobes methods relying on cascading strategy, where base models outputs are iteratively included once at a time, as new features in the training set. The third group is based on the stacking paradigm [33]. Using stacking, most often a meta-learning approach is employed to combine the available forecasts. This method implicitly learns an adequate combination by modelling inter-dependencies between models.

Another key point in learning ensemble for time series data is to be able to cope the time-evolving nature of data. This can be achieved by building dynamic ensemble selection frameworks through adaptive ensemble constructions on different levels (i.e. base models selection, base models/ensemble parameters adaption, blind/informed retraining).

In this paper, we propose a dynamic ensemble selection framework that operates on two main ensemble construction stages: pruning and integration. Given a pool of candidate base models, the first stage of the framework is motivated by the fact that due to the time-evolving nature of the data structure, base models performance changes over time. This performance is also subject to concept-drifts, when considering the relation between the output predictions of base models and the target time series. A drift detection mechanism is employed to exclude models whose performance becomes significantly poor compared to the remaining models and to identify the top base models in terms of performance. Performance is assessed in this context using a custom measure based on the Pearson's correlation (i.e. commonly used to deal with time series data [27]) between base models forecasts and the target time series on a sliding window validation set. After each drift detection, the top base models are identified.

Since diversity is a fundamental component in ensemble methods [4], we propose a second stage selection through clustering model outputs. Clusters and top base models are updated after each drift detection. At each cluster computation, the models that belong to the cluster representatives are selected. Finally, the outputs of the selected models are combined together using a voting strategy based on a sliding-window weighted average. Our framework is denoted in the rest of the paper, **DEMSC**: Drift-based Ensemble Member Selection using Clustering. DEMSC is illustrated in Fig. 1.

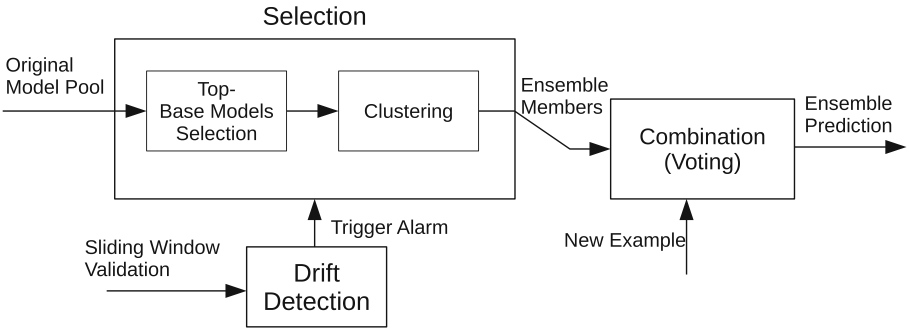

Fig. 1. Components of our method DEMSC

We validate our framework using 16-real world time series data sets. Different variations of our method have been carried out to assess the impact of each stage (i.e. component) by changing the clustering method or the combination rules (i.e. for example, using stacking nstead of voting). Empirical results suggest that our method outperforms traditional state-of-art methods for ensemble learning and other metalearning approaches such as stacking [33] and bagging [3] and is competitive with adaptive approaches for dynamic ensemble selection [6]. We note that all experiments are fully reproducible. Both code and datasets are publicly available in this repository.[1]

The main contributions of this paper are as follows:

- A drift-based dynamic selection ensemble framework is introduced. Opposingly to existing dynamic ensemble selection methods which rely on continuous updates (i.e. blindly at each time instant or periodically), our selection is automatically and adaptively performed in an **informed manner** based on models performance drift detection mechanism.
- An online clustering approach in combination with a pre-selection is applied for model selection. The model selection is triggered by the drift detection mechanism.
- The framework is devised to work in an automated fashion, in the sense that a sliding-window validation set is used for the drift inspection. The data from

[1] https://github.com/AmalSd/DEMSC.

the validation set is used as input for the clustering. The clustering method also optimizes the number of clusters.
- A comparative empirical study with S.o.A ensemble methods and different variations of our framework, including a discussion about their implications in terms of predictive performance and computational resources, is conducted.

In the remainder of this paper, we describe the proposed approach, after discussing related work. Then, we present an exhaustive experimental evaluation of its efficiency and scalability. Finally, the last section concludes the paper.

2 Related Works

Ensembles for time series forecasting have always attracted the attention of the machine learning community [18,26]. More precisely, methods for dynamically combining models outputs, using both windowing strategies [26,28] and metalearning approaches [6,7] have particularly intrigued the interest of the community over the last few years. In this section, we briefly describe the state-of-the-art methods within these approaches. We list their characteristics and limitations and we highlight our contributions.

Combination in an ensemble can be made using the average of the available base models' outputs. This was commonly applied not only in forecasting [8], but also in regression [22]. Simple averages can be enhanced by the use of model selection before aggregation, this approach is known as trimmed means [21]. To deal with time-evolving data, one of the most successful approaches is to compute weighted averages over a time-sliding window, by setting up the weights to be proportional to some performance measures [26,28]. A forgetting mechanism can be employed to the time window to increase the impact of recent observations (i.e most recent performance of the predictive models).

Our method uses the weighted sliding-window average for the combination of the base models' predictions to produce the final forecast value. However, we have tested different variations of our method by replacing this combination strategy with metalearning. More details are provided in Sect. 3.

Metalearning as a way for modeling the learning process of a learning algorithm, can be exploited in both pruning and combination stages. Most of existing works followed this approach to learn models combination (i.e. integration) rules given the set of model forecasts [15,30,33].

Furthermore, combination can be performed dynamically so that its rules are changing over time. A popular approach for dynamically combining experts is to apply multiple regression on the output of the experts. For example, Gaillard and Goude [15] use Ridge regression to adaptively estimate the weights that are assigned to the base-learners. Another recent approach which has successfully exploited metalearning based on arbitrating [6], which was originally introduced for the combination of classifiers. Arbitrated Dynamic Ensemble (ADE) [7] uses meta-learning for estimating each candidates errors and selects members based on the estimated errors. ADE uses Random Forest as meta-learner. For each candidate, a separate meta-model is trained. The selected members are combined with a convex combination. Each candidate weight is based on the estimated errors combined with a softmax. The weights are additionally adapted to take diversity between ensemble members into account.

Our framework exploits metalearning in the pruning stage where a dynamic clustering of base models is performed. Only the cluster representatives are selected to take part in the integration phase. The idea of model clustering was introduced in [28] to cluster a pool of base models into main families to enhance diversity. However, in [28] the clustering was performed offline and kept static in the online phase. Only cluster representatives changed over time in [28]. Oppositely, in our framework, model clusters are recomputed each time dependencies between base models and the target series change significantly in the current sliding-window validation set. This **informed** ability is a key point in our method and to the best of our knowledge this is the first approach to perform dynamic ensemble selection adaptively following a drift in models performance detection mechanism. Oppositely, most of existing methods for dynamic selection keep the learned meta-model static (i.e. no meta-learner retraining is performed) in the testing phase and only few works state the advantage of performing periodic retraining in a blind manner [7] (i.e. just setting up a fixed period for the meta-learner retraining).

3 Methodology

This Section introduces **DEMSC** and its three basic components: (i) First, we describe the drift-based pre-selection step to get the top base models in terms of performance; (ii) The second stage consists of first clustering the top base models and select one representative model for each cluster; (iii) Finally, each selected model's output is combined in a weighted average where the weights are inversely proportional to the model recent loss.

3.1 Problem Formulation

A time series X is a temporal sequence of values $X = \{x_1, x_2, \cdots, x_t\}$, where x_i is the value of X at time i. Typical solution for time series forecasting include traditional univariate time series analysis models, such as the popular Box-Jenkins ARIMA family of methods [2] or exponential smoothing methods [20]. Typical regression models can be applied in the context of forecasting by using a time delay embedding which maps a set of target observations to a K-dimensional feature space corresponding to the K past lagged values of each observation [6].

Denote with $P_M = \{M_1, M_2, \cdots, M_N\}$ the pool of trained base forecasting models. Let $\hat{x} = (\hat{x}_{M_1}, \hat{x}_{M_2}, \cdots, \hat{x}_{M_N})$ be the vector of forecast values of X at time instant $t+1$ (i.e. x_{t+1}) by each of the base model in P_M. The goal of the dynamic selection is identifying which $\hat{x}_{M_i}$ values should be integrated in the weighted average.

To do so, a two-staged selection procedure is devised. The first stage is a pre-selection stage which aims to keep only accurate model forecasts using a model performance drift detection. This stage discards models with poor performance whose forecasts inclusion in the ensemble would deteriorate the forecasting accuracy. This deterioration is more perceptible using simple average for integration and can be covered to some extent using a weighting strategy. The second stage aims to enhance diversity aspect with the use of clustering.

3.2 A Drift-Based Model Pre-selection

The drift-based time series selection was first applied in [28] in the context of spatio-temporal features selection to be the input of a multivariate autoregressive model. Similarly, we can treat the set of base models forecasts as a set of explanatory variables or causes to our target time series. To do so, dependencies between the set of base model forecasts and target time series can be continuously computed and monitored over a sliding-window validation set. Suppose we want to compute the prediction for time instant $t+1$, the validation sliding-window of size W over X is defined by the sequence $X_{W,t} = \{x_{t-W+1}, x_{t-W+2}, \cdots, x_t\}$. Let $\hat{X}^{M_i}_{W,t} = \{\hat{x}^{M_i}_{t-W+1}, \hat{x}^{M_i}_{t-W+2}, \cdots, \hat{x}^{M_i}_{t}\}$ be the predicted sequence of values by the model M_i on $X_{W,t}$, where $M_i \in P_M$.

A subset K of highly correlated models with the target, denoted "top-base" models, are selected using a sliding-window similarity measure computed on $X_{W,t}$. K is a user-defined hyperparameter. Hereby, we propose to use a custom measure based on the Pearson's correlation - commonly used to deal with time series data [27]-denoted as *SRC* - Scaled Root Correlation and defined as:

$$corr(\hat{X}^{M_i}_{W,t}, X_{W,t}) = \frac{\tau - \frac{\sum_{j=1}^{W} \hat{x}^{M_i}_{t-W+j} \sum_{j=1}^{W} x_{t-W+j}}{W}}{\sqrt{\sum_{j=1}^{W} (\hat{x}^{M_i}_{t-W+j})^2 - \frac{(\sum_{j=1}^{W} \hat{x}^{M_i}_{t-W+j})^2}{W}} \sqrt{\sum_{j=1}^{W} (x_{t-W+j})^2 - \frac{(\sum_{j=1}^{W} x_{t-W+j})^2}{W}}} \tag{1}$$

$$SRC(\hat{X}^{M_i}_{W,t}, X_{W,t}) = \sqrt{\frac{1 - corr(\hat{X}^{M_i}_{W}, X_W)}{2}} \in [0, 1] \tag{2}$$

where $\tau = \sum_{j=1}^{W} \hat{x}^{M_i}_{t-W+j} x_{t-W+j}$. Naturally, with time-evolving data, dependencies change over time and follow non-stationary concepts. Stationarity in this context can be formulated as follow:

Definition 1 (Weak stationary Dependencies). *Let $C_t \in \mathbb{R}^{N \times N}$ be a resulting symmetric similarity matrix between the base models and the target time series over W (i.e. derived from the above similarity metric), where $N = |P_M|$ and c_t be a vector containing all the elements in C_t where $c_{j,t} \geq c_{j-1,t}, \forall j \in \{1 \ldots N^2\}$. Let μ denote the minimum* SRC *coefficient of P_M at the initial instant of its generation t_i. The dependence structure is said to be weakly stationary if the true mean of Δc_t is 0:*

$$\Delta c_t = \left| c_{1,t} - \mu \right| \tag{3}$$

Following this definition, we can assume that the distance between the two most dissimilar random processes within the same pool of models sets its boundary under a form of a logical *diameter*. If this boundary diverges in a significant way over time, a drift is assumed to take place. We propose to detect the validity of such assumption using the well-known Hoeffding Bound [19], which states that after W independent observations of a real-value random variable with range R, its true mean has not diverged if the sample mean is contained within $\pm\epsilon_F$:

$$\epsilon_F = \sqrt{\frac{R^2 \ln(1/\delta)}{2W}} \tag{4}$$

with a probability of $1 - \delta$ (a user-defined hyperparameter). Once the condition of the *weak stationary dependencies* presented in Definition 1 is violated, an alarm is triggered, the top base models using C_t are updated. Afterwards, the dependency monitoring process is continued by sliding the time window for the next prediction and the reference *diameter* μ is reset by setting $t_i = t$.

3.3 Model Clustering

One of the most important aspects for successful ensembles is diversity [4,7,28]. Typically, this diversity is initially reflected in the distinctive patterns of each base learner's inductive bias derived from the different hypothesis on which each base learner is built to model the input data and its dependence structure. Surprisingly, the enforcement and evaluation of diversity on ensembles for time series data is still a quite unexplored topic-especially for forecasting problems [23,28]. However, the expected error decomposition for ensemble schemata [4,31] in general helps to get an intuition about the importance of diversity. More precisely, the expected error can be decomposed into *bias*, *variance* and *covariance*.

In DEMSC, we propose a second-stage selection that tries to ensure such diversity through *clustering*. Predictions of K top-base models on the time sequence $X_{W,t}$, are considered as W-dimensional vector features to cluster the models. To compute clusters for time series, several techniques are proposed in literature such as K-means and hierarchical clustering [1]. However, one of the main issues presented by time series clustering is the choice of similarity/distance measure as most of typical distance measures such as the Euclidean distance do not take dependence structures of time series data into account [1]. To overcome this issue, we used an improper maximum likelihood estimator based clustering method (IMLEC) [9], which is based on a multivariate Gaussian model where parameters are estimated using Expectation Maximization algorithm [25].

This method has the advantage over Euclidean Distance (ED)-based clustering methods by contributing to the reduction of the covariance term of the ensemble error and thus to the reduction of the overall error. For instance, ED-based clustering methods like K-Means, do not take into account the covariance of the data. If we consider two candidate time series that have dependence over a high number of components of their W-dimensional feature space (i.e. high covariance is assumed to take place), the probability of attributing them to the same cluster by fitting the adequate parameters of the Gaussian mixture to the data is higher than simply using an ED-based method, which would probably assign them to different clusters based on their closeness to the current cluster centres. As a results, models belonging to different clusters have more likely low covariance. Therefore, the final step in the selection consists of selecting one representative model for each cluster. We simply select the closest model to each cluster center.

3.4 Model Combination

The final selected base-models are integrated using a sliding-window weighted average [26,28]. Let P_{M^f} be the pool of final selected base models to take part in the ensemble for the prediction of time instant $t+1$ and $\hat{x}_{j,t+1}$ the output of model M_j in time instant $t+1$. The final prediction is obtained by:

$$\hat{x}_{t+1} = \sum_{j=1}^{|P_{M^f}|} \frac{\left[\left(1-\chi_{j,t}\right)\hat{x}_{j,t+1}\right]}{\sum_{j=1}^{|P_{M^f}|}\left(1-\chi_{j,t}\right)} : \chi_{j,t} \in [0,1], \forall j, t \qquad (5)$$

where $\chi_{j,t}$ is a normalized version of the recent loss of the model M_j on $[t-W+1, t]$ on the random process which computation is given by an evaluation metric of interest (i.e Normalized Root Mean Square Error (NRMSE) in our case).

This methodology was exhaustively tested over data collected from 16 real-world datasets. Further details are provided in the following Section.

4 Experiments

In this section, we present the experiments carried out to validate DEMSC and to answer the following research questions:

Q1: How is the performance of DEMSC compared to the state-of-the-art methods for time series forecasting tasks and to existing dynamic ensemble selection approaches?
Q2: What is the advantage of the performance drift detection mechanism, which triggers the ensemble members pre-selection, in terms of accuracy?
Q3: What is the impact of clustering and how does the IMLEC-clustering perform compared to commonly used clustering strategies for time series data?
Q4: What is the impact of different combination strategies on the performance?
Q5: Is there an advantage in terms of computational resources if the ensemble members selection is done in an **informed** fashion (i.e. only triggered by the drift detection alarm)?

4.1 Experimental Setup

The methods used in the experiments were evaluated using RMSE. In each experiment, the time series data was split to 75% for training, and 25% for testing. The results are compared using the non-parametric Wilcoxon Signed Rank test. We used 16 real-world time series shown in Table 1 for our experiments. An embedding dimension of 5 was used for all the time series.

Table 1. List of datasets used for the experiments.

ID	Time series	Data source	Data characteristics
1	Water consumption	Oporto city [7]	Daily obs. Jan, 2012–Oct, 2016
2	Temperature	Bike sharing [7]	
3	Feeling temperature		Hourly values from Jan. 1, 2011 to Mar. 01, 2011
4	Humidity		
5	Windspeed		
6	Registered users		
7	Total bike rentals		
8	Global horizontal radiation	Solar radiation monitoring [7]	Hourly values Feb. 16, 2016–May 5, 2016
9	Direct normal radiation		
10	Diffuse horizontal radiation		
11	Vatnsdalsa	River flow [7]	Daily observations from Jan. 1, 1972 to Dec. 31, 1974
12	Jokulsa Eystri		
13	Chill temperature	Weather data [29]	Hourly observations from Apr. 25, 2016 to Aug. 25, 2016
14	Total cloud cover		
15	Wind speed		
16	Precipitation		

4.2 Ensemble Setup and Baselines

There is no forecasting method that performs best on every time series. For our candidate pool, we inconstantly used and tested different families of models:

GBM Gradient Boosting Machine [12];
SVR Support Vector Regression [11];
PPR Projection Pursuit Regr. [13];
PCMR [24] Principal Component Regr.
PLS [24] Partial Least Squares Regr.
GP Gaussian Process [32]
RFR Random Forest [3]
MARS MARS [14];
DT Decision Tree Regr.;
MLP [17] Multilayer Perceptron

Different parameter settings for the models, generate a pool of 30 candidate models, that we will use for the ensemble methods. We can see in Fig. 2, that the forecasting methods have a high variance as their performance changes across the different time series.

There is no clear best performing model. This motivates the dynamic combination of different forecasting models to an ensemble.

DEMSC has a number of hyperparameters that are summarized in Table 2. For IMLEC-clustering, we used the R-package of the authors of [9]. The maximum number of cluster is a user-defined parameter. However, it can be automatically reduced by removing outliers and noisy data that cannot be fitted to any cluster.

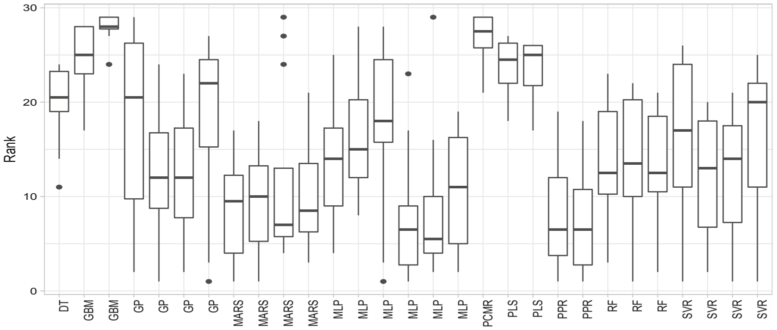

Fig. 2. Distribution of rank of the base models across the 16 time series (similar models names in the x-axis have different parameters)

Table 2. Hyperparameter of DEMSC and their chosen values for the experiments.

Parameter	Value
Number of top base models	Half of candidate pool
Maximum number of clusters	Half of top base models
Hoeffding-bound δ	0.95
Size of sliding window W	A user-defined hyperparameter

We compare the performance of DEMSC against the following approaches:

RF [3]: Random Forest uses bagging to create an ensemble of regression trees.
GBM [12]: Gradient boosting machine that uses boosting to create an ensemble of regression trees.
SE [8]: A static ensemble that averages the performance of all base learners using arithmetic mean.
SWE [26]: A linear combination of the base learners predictions. The weights are based upon recent performance over a time sliding-window.
ARIMA [2]: ARIMA model for time series forecasting.
EWA [16]: Forecasting combination with exponential weighted averages.
FS [16]: The fixed share approach from Herbster and Warmuth, which is designed for tracking the best expert across a time series.
OGD [16]: An approach based on online gradient descent that provides theoretical loss bound guarantees.
MLPOL [16]: A polynomially weighted average forecast combination.
Stacking [33]: An approach for ensembles using Random Forest as metalearner.
DETS [5]: An advanced version of SWE, selecting a subset of members based on recent errors and uses a smoothing function on the average of recent errors for weighting.
ADE [6]: Uses a Random Forest for estimating each candidates errors and selects member based on the estimated errors. Weighting is also based on the

estimated errors combined with a softmax. Weights are additionally adapted to take diversity between ensemble members into account.

We also compare DEMSC with some variants of itself. All of these variants except one, use the sliding window ensemble for combining the ensemble members predictions.

DEMSC-NoSel: Same as our method but without the Top-Base Models selection. Clusters are updated periodically.

Top-Base Models: Only the pre-selection of the Top-Base models based on correlation (no clustering is applied afterwards)

DEMSC-kMeans: The clustering method is replaced with K-Mean with ED distance (K is tuned using the average silhouette method).

DEMSC-DTW: The clustering method is replaced with dynamic time warping clustering.

DEMSC-stacking: The stacking variant differs from our method only in the combination step. Instead of a sliding window ensemble, a stacking approach is used in this variant (**PLS** is used as metalearner).

4.3 Results

Table 3 presents the average ranks and their deviation for all methods. For the paired comparison, we compare our method DEMSC against each of the other methods. We counted wins and losses for each dataset using the RMSE scores. We use the non-parametric Wilcoxon Signed Rank test to compute significant wins and losses, which are presented in parenthesis (i.e. the significance level = 0:05). Figure 3 presents the distribution of ranks across the different time series for all methods.

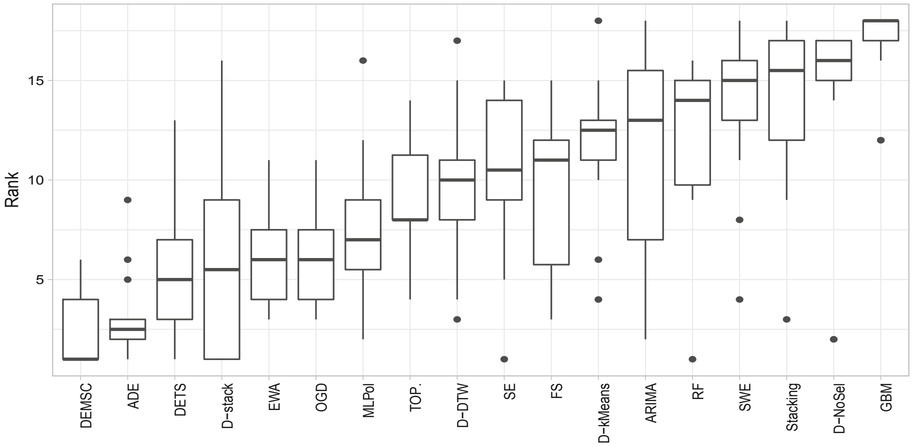

Fig. 3. Distribution of the ranks of ensemble methods across the different time series, D is used as abbreviation for DEMSC

Table 3. Paired comparison between DEMSC and different baseline methods for 16 time series. The rank column presents the average rank achieved by each model and the standard deviation of the rank across the different time series. A rank of 1 means the model was the best performing on all time series.

Method	Our method		
	Losses	Wins	Avg. rank
RF	1(0)	15(13)	12.2 ± 3.7
GBM	0(0)	16(16)	17.0 ± 2.0
SE	1(0)	15(13)	10.4 ± 3.6
SWE	0(0)	16(14)	13.8 ± 3.5
ARIMA	1(1)	15(11)	11.5 ± 5.1
EWA	1(0)	15(12)	6.3 ± 2.6
FS	1(0)	15(13)	9.5 ± 3.6
OGD	1(0)	15(12)	6.3 ± 2.5
MLPOL	2(1)	14(10)	7.2 ± 3.5
Stacking	1(1)	15(14)	13.9 ± 3.8
ADE	7(7)	7(3)	3.1 ± 2.0
DETS	4(4)	7(4)	5.4 ± 2.8
DEMSC-NoSel	0(0)	16(14)	13.9 ± 4.7
Top-Base Models	1(0)	15(14)	9.2 ± 2.9
DEMSC-kMeans	0(0)	16(16)	11.7 ± 3.1
DEMSC-DTW	1(0)	15(14)	9.6 ± 3.5
DEMSC-stacking	5(2)	9(6)	5.7 ± 4.3
DEMSC	–	–	2.3 ± 1.7

DEMSC has advantages over the compared methods except for ADE. The approaches for combining individual forecasters, which are SE, SWE, OGD, FS, EWA and MLPOL, show a big difference in the average rank compared to DEMSC. ARIMA, a state-of-the-art method for forecasting, has a big difference in the average rank as well. Common ensemble methods like RF, GBM, OGD and Stacking, compare poorly to all methods specialized for combining forecasters. The two competitive approaches to our method are ADE and DETS, with DETS having a higher average rank but performing well in the pairwise comparison. ADE is competitive to DEMSC and have a higher average rank, but it is comparable to DEMSC in terms of wins and losses. Looking at the distribution of ranks in Fig. 3, we see that ADE has some clear outliers, while being robust in the other cases. DEMSC is within the range of the first 4 ranks and has a median of 1 with no clear outliers.

To further investigate the differences in the average ranks, we use the post-hoc Bonferroni-Dunn test [10] to compute critical differences. We present the critical differences between the methods relative to each other in Fig. 4. Adding to the results of Table 3, we note critical differences between DEMSC and most of the other methods, with the exceptions of ADE, DETS, EWA, ODG and MLPOL. We already discussed the comparable performance of DEMSC and ADE. Both methods share the critical differences to other methods. Regarding research question **Q1**, our results show that DEMSC is competitive with ADE and outperforms other combination approaches for time series forecasting. The average rank of DEMSC is better than ADE, but we do not see the main advantage of our method in the performance but more in the complexity and computational requirements, which we will discuss later.

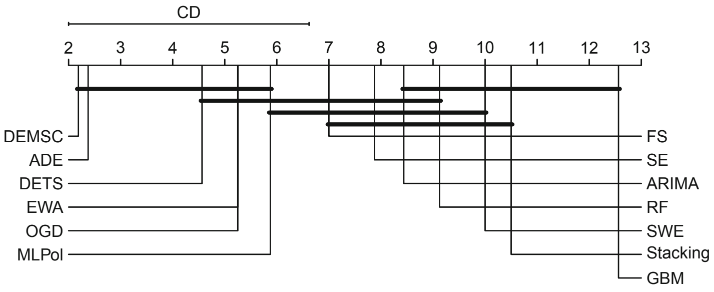

Fig. 4. Critical difference diagram for the post-hoc Bonferroni-Dunn test, comparing DEMSC with the other baseline ensemble methods.

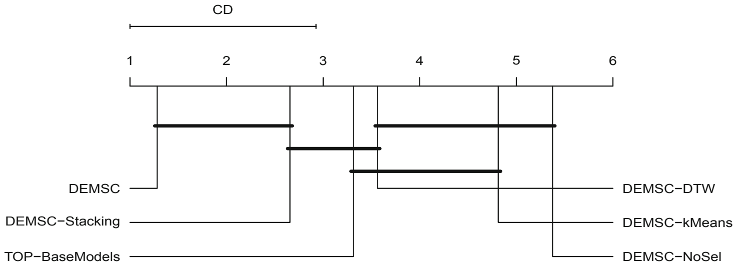

Fig. 5. Critical difference diagram for the post-hoc Bonferroni-Dunn test, comparing DEMSC against variants of our method.

Comparing DEMSC to different variants of our method, we see a clear advantage in using all the presented components. The results are the worst, if we only use clustering for selection (DEMSC-NoSel). The pre-selection is needed to ensure that the clustering uses the set of the most accurate models. Using only the top-Base Models, gives a useful performance, but we show that it can be further improved using clustering. Comparing the three different clustering methods (DEMSC-kMeans, DEMSC-DTW, DEMSC(IMLEC)) we can see the clear advantage of the IMLEC-clustering. Using a time warping clustering gives a slight improvement over K-means, but both of them do not improve the preselection. Using IMLEC-clustering improves the performance drastically. This can be explained partially by enhancing the diversity aspect discussed in Sect. 3. We see that the combination part of our method has a small impact. Both variants using IMLEC-clustering, with either sliding window ensemble or stacking for the combination, have a clear advantage over the other clustering variants (see question **Q3**). We present in Fig. 5 the critical differences of the methods regarding the average rank. The only variant, where the difference in average rank to DEMSC is not critically different is the stacking variant (DEMSC-stacking). The stacking variant's average rank is higher. This answers the research question **Q4** regarding the impact of each component of the method.

We can also answer research question **Q2**, asking about the impact of using a drift detection to trigger the member selection. Performance wise, we see that our method performs on the same level or even slightly better than the best state-of-the-art approach. The motivation for using a drift detection is to update the ensemble only when necessary. This should result in faster predictions and less computational requirements (see question **Q5**). We see in Table 4 that the average runtime of ADE is more than twice as long as the runtime of our method. The high deviation of the runtime of our method is due to the different datasets, that have more or less drifts detected.

Table 4. Empirical runtime comparison between DMESC and the most competitive state-of-the-art method (ADE).

Method	Avg. runtime in sec.
DEMSC	66.39 ± 26.4
ADE	156.97 ± 18.3

4.4 Discussion

We presented results that empirically show that DEMSC has performance advantages compared to other ensemble methods for forecasting and is competitive with the most recent state-of-the-art approaches for dynamically combining forecasting methods.

We show that our method, using a combination of clustering and a performance based pre-selection, is able to perform on a high level. The pre-selection assures that only accurate models are used in the ensemble. The clustering groups similar models based on their predictions. We then select clusters representatives. This leads to an ensemble with accurate and diverse members, which has been theoretically shown to be required for an ensemble to outperform it's members. Neither of the parts can reach state-of-the-art performance on its own, but the combination makes them very powerful.

The usage of drift detection enables our method to construct a new ensemble given changes in the nature of the dependencies between base-models and the target time series. If there is no change, then there is also no need to construct a new ensemble. Therefore, the drift detection reduces the computations.

DEMSC method and the complex meta-learning approach ADE perform on the same level. To reach same performance, we only need pre-selection and clustering, triggered by a drift detection. Compared to ADE, which needs to train a meta-model for each candidate, our method is computationally cheaper. For the experiments a prediction with ADE needed on average twice as long as our method.

5 Final Remarks

This paper introduces DEMSC: a novel, practically useful dynamic ensemble members selection framework for time series forecasting. DEMSC uses a two-staged selection procedure which on the one hand enhances accuracy by performing informed selection of base learners at test time based on a base models performance drift detection mechanism and diversity on the other hand through an online clustering approach. An exhaustive empirical evaluation, including 16 real-world datasets and multiple comparison algorithms shows the advantages of DEMSC. As a future work, we aim to add a drift-informed procedure for retraining the base-learners.

Acknowledgments. This work is supported by the Deutsche Forschungsgemeinschaft (DFG) within the Collaborative Research Center SFB 876 and the Federal Ministry of Education and Research of Germany as part of the competence center for machine learning ML2R (01S18038A).

References

1. Aghabozorgi, S., Shirkhorshidi, A.S., Wah, T.Y.: Time-series clustering–a decade review. Inf. Syst. **53**, 16–38 (2015)
2. Box, G.E., Jenkins, G.M., Reinsel, G.C., Ljung, G.M.: Time Series Analysis: Forecasting and Control. Wiley, San Francisco (2015)
3. Breiman, L.: Bagging predictors. Mach. Learn. **24**(2), 123–140 (1996)
4. Brown, G., Wyatt, J.L., Tiňo, P.: Managing diversity in regression ensembles. J. Mach. Learn. Res. **6**(2), 1621–1650 (2005)
5. Cerqueira, V., Torgo, L., Oliveira, M., Pfahringer, B.: Dynamic and heterogeneous ensembles for time series forecasting. In: IEEE International Conference on Data Science and Advanced Analytics (DSAA), pp. 242–251. IEEE (2017)
6. Cerqueira, V., Torgo, L., Pinto, F., Soares, C.: Arbitrated ensemble for time series forecasting. In: Ceci, M., Hollmén, J., Todorovski, L., Vens, C., Džeroski, S. (eds.) ECML PKDD 2017. LNCS (LNAI), vol. 10535, pp. 478–494. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-71246-8_29
7. Cerqueira, V., Torgo, L., Pinto, F., Soares, C.: Arbitrage of forecasting experts. Mach. Learn. **108**(6), 913–944 (2018). https://doi.org/10.1007/s10994-018-05774-y
8. Clemen, R.T., Winkler, R.L.: Combining economic forecasts. J. Bus. Econ. Stat. **4**(1), 39–46 (1986)

9. Coretto, P., Hennig, C.: Robust improper maximum likelihood: tuning, computation, and a comparison with other methods for robust gaussian clustering. J. Am. Stat. Assoc. **111**(516), 1648–1659 (2016)
10. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. **7**, 1–30 (2006)
11. Drucker, H., Burges, C.J., Kaufman, L., Smola, A.J., Vapnik, V.: Support vector regression machines. In: Advances in Neural Information Processing Systems, pp. 155–161 (1997)
12. Friedman, J.H.: Greedy function approximation: a gradient boosting machine. Ann. Stat. **29**, 1189–1232 (2001)
13. Friedman, J.H., Stuetzle, W.: Projection pursuit regression. J. Am. Stat. Assoc. **76**(376), 817–823 (1981)
14. Friedman, J.H., et al.: Multivariate adaptive regression splines. Ann. Stat. **19**(1), 1–67 (1991)
15. Gaillard, P., Goude, Y.: Forecasting electricity consumption by aggregating experts; how to design a good set of experts. In: Antoniadis, A., Poggi, J.-M., Brossat, X. (eds.) Modeling and Stochastic Learning for Forecasting in High Dimensions. LNS, vol. 217, pp. 95–115. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-18732-7_6
16. Gaillard, P., Goude, Y.: opera: Online Prediction by Expert Aggregation (2016). https://CRAN.R-project.org/package=opera. r package version 1.0
17. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016). http://www.deeplearningbook.org
18. Gooijer, J.G.D., Hyndman, R.J.: 25 years of time series forecasting. Int. J. Forecast. **22**(3), 443–473 (2006)
19. Hoeffding, W.: Probability inequalities for sums of bounded random variables. In: Fisher, N.I., Sen, P.K. (eds.) The Collected Works of Wassily Hoeffding, pp. 409–426. Springer, New York (1994). https://doi.org/10.1007/978-1-4612-0865-5_26
20. Hyndman, R.J., Koehler, A.B., Snyder, R.D., Grose, S.: A state space framework for automatic forecasting using exponential smoothing methods. Int. J. Forecast. **18**(3), 439–454 (2002)
21. Jose, V.R.R., Winkler, R.L.: Simple robust averages of forecasts: some empirical results. Int. J. Forecast. **24**(1), 163–169 (2008)
22. Khiari, J., Moreira-Matias, L., Shaker, A., Ženko, B., Džeroski, S.: MetaBags: bagged meta-decision trees for regression. In: Berlingerio, M., Bonchi, F., Gärtner, T., Hurley, N., Ifrim, G. (eds.) ECML PKDD 2018. LNCS (LNAI), vol. 11051, pp. 637–652. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-10925-7_39
23. Krawczyk, B., Minku, L.L., Gama, J., Stefanowski, J., Woźniak, M.: Ensemble learning for data stream analysis: a survey. Inf. Fusion **37**, 132–156 (2017)
24. Mevik, B.H., Wehrens, R., Liland, K.H.: PLS: Partial Least Squares and Principal Component Regression (2018). https://CRAN.R-project.org/package=pls
25. Moon, T.K.: The expectation-maximization algorithm. IEEE Signal Process. Mag. **13**(6), 47–60 (1996)
26. Moreira-Matias, L., Gama, J., Ferreira, M., Mendes-Moreira, J., Damas, L.: Predicting taxi-passenger demand using streaming data. IEEE Trans. Intell. Transp. Syst. **14**(3), 1393–1402 (2013)
27. Rodrigues, P.P., Gama, J., Pedroso, J.: Hierarchical clustering of time-series data streams. IEEE Trans. Knowl. Data Eng. **20**(5), 615–627 (2008)
28. Saadallah, A., Moreira-Matias, L., Sousa, R., Khiari, J., Jenelius, E., Gama, J.: Bright-drift-aware demand predictions for taxi networks. IEEE Trans. Knowl. Data Eng. **32**, 234–245 (2018)

29. Stoffel, T., Andreas, A.: NREL solar radiation research laboratory (SRRL): Baseline measurement system (BMS); Golden, Colorado (data), July 1981
30. Todorovski, L., Džeroski, S.: Combining classifiers with meta decision trees. Mach. Learn. **50**(3), 223–249 (2003)
31. Ueda, N., Nakano, R.: Generalization error of ensemble estimators. In: 1996 IEEE International Conference on Neural Networks, no. xi, pp. 90–95 (1996)
32. Williams, C.K., Rasmussen, C.E.: Gaussian Processes for Machine Learning, vol. 2. MIT Press, Cambridge (2006)
33. Wolpert, D.H.: Stacked generalization. Neural Netw. **5**(2), 241–259 (1992)