


# Online Adaptive Multivariate Time Series Forecasting<sup>\*</sup>

Amal Saadallah , Hanna Mykula, and Katharina Morik

Artificial Intelligence Group, Department of Computer Science, TU Dortmund,  
Dortmund, Germany

{amal.saadallah, hanna.mykula, katharina.morik}@tu-dortmund.de

**Abstract.** Multivariate Time Series (MTS) involve multiple time series variables that are interdependent. The MTS follows two dimensions, namely spatial along the different variables composing the MTS and temporal. Both, the complex and the time-evolving nature of MTS data make forecasting one of the most challenging tasks in time series analysis. Typical methods for MTS forecasting are designed to operate in a static manner in time or space without taking into account the evolution of spatio-temporal dependencies among data observations, which may be subject to significant changes. Moreover, it is generally accepted that none of these methods is universally valid for every application. Therefore, we propose an online adaptation of MTS forecasting by devising a fully automated framework for both adaptive input spatio-temporal variables and adequate forecasting model selection. The adaptation is performed in an informed manner following concept-drift detection in both spatio-temporal dependencies and model performance over time. In addition, a well-designed meta-learning scheme is used to automate the selection of appropriate dependence measures and the forecasting model. An extensive empirical study on several real-world datasets shows that our method achieves excellent or on-par results in comparison to the state-of-the-art (SoA) approaches as well as several baselines.

**Keywords:** Multivariate Time Series · Forecasting · Automated Model Selection · Spatio-temporal Dependencies · Concept-drift.

## 1 Introduction

Time series forecasting is an important task in time series analysis to study the behavior of temporal data and forecast its future values [21, 20]. It is widely applied in various fields, including weather forecasts, energy demand/consumption predictions, and stock market prices forecasting, to name but a few [19–21]. In nowadays’ rapidly growing digital environments and Internet-of-Things systems,

---

<sup>\*</sup> 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 (01—S18038A).

the representation of time series data often involves multiple interdependent variables, thus creating Multivariate Time Series (MTS) data [26]. On the one hand, this data represents an enriched form of information about the application. On the other hand, the number of these variables can increase drastically and might include irrelevant and redundant ones. This may heighten the curse of dimensionality. Therefore, it is necessary to carefully select the most important time series variables. The evolution of MTS is spatio-temporal, along with the different variables and over time, respectively. However, this spatio-temporal data may involve multiple non-stationary processes and the dependencies along its composing variables may also follow a non-stationary process. As a result, the relationship between some time series variables and the target one might change significantly over time. This phenomenon is broached in the machine learning literature as concept drift [8]. Hence, previously learned concepts about data become no longer valid, making the offline input variable selection procedures inappropriate for making future predictions. Therefore, the selection of time series variables should cope with the evolving nature of the spatio-temporal dependencies in the MTS data.

Various Machine Learning (ML) models have already been successfully applied to solve the forecasting task either by dealing with the MTS data as a collection of ordered sequences of observations in an offline [26] or a streaming fashion [20], or by using an embedding of the MTS to reformulate the forecasting task as a regression task [21]. However, it is generally accepted that none of the ML methods is universally valid for every task, in particular for forecasting [21]. Therefore, in addition to adaptive time-dependent spatial variables selection, adequate model selection is required to cope with the characteristics of the MTS. Most of the existing MTS forecasting models operate in a static manner, i.e. the model is trained offline using some collection of historical data and a fixed selection of input variables. Its parameters are optimized once at training time. At test time, the model is deployed with fixed learned parameters and fixed information about temporal and spatial data [26]. Methods for online MTS forecasting focus either on very specific application/setting [9] or use a very specific family of ML models, such as Deep Neural Networks [27, 22]. More recently, a drift-aware Vector Autoregressive (VAR) model has been proposed in [20]. In contrast to the classic VAR model which takes as input, all the variables of the MTS [26], an adaptive selection procedure of a subset of input variables is done in the drift-aware VAR. The update of this subset depends on a change in the Pearson-Correlation (PC) [16] measured between two variables over a time-sliding window, which we assume has occurred due to concept drift. Even though the proposed method is online and adaptive, it focuses only on a particular model, namely the VAR. In addition, the time series variables selection is done by ranking them according to their relevance to the target time series using PC. No further analysis is carried out to investigate the redundancy in the selected subset. The relevance/similarity to the target is measured using the PC coefficient. However, it has been proven that there is no single universal measure for the similarity between two time series either for relevance or redundancy analysis [16]. The quality of the achieved

results in this context depends to a large extent on the used time series measure [16].

In this work, we propose an online adaptive framework for MTS forecasting which performs both input variable time series selection and adequate forecasting model selection. Input variables selection is done on two stages using relevance and redundancy analysis. The selection is made dynamically and adaptively in an informed manner following concept drift detection. The concept drift detection covers the two MTS dimensions, namely spatial and temporal. Spatial dependencies indicate the similarity between the input variables at one time instant. We monitor the change of the similarity values over time. Temporal dependencies indicate the patterns discovered within the same spatial dimension over time. The drift detection within the temporal dimension is ensured by tracking the change in the estimated model’s performance on a given target time series variable over time. In addition, the choice of the adequate relevance and redundancy measures, as well as the forecasting model is done in an automated fashion using meta-learning on well-devised MTS meta-features. Our framework is denoted in the rest of the paper, OAMTS: Online Adaptive Multivariate Time series forecasting. We further conduct a comprehensive empirical analysis to validate our method using 66 real-world MTS datasets from different domains. We have created separate meta-data which cover a collection of real-world and synthetic MTS with various characteristics for the meta-learning task. The obtained results show that our method achieves excellent results in comparison to the SoA approaches for MTS forecasting. We note that all experiments are fully reproducible and that both the code and datasets are publicly available <sup>1</sup>.

The main contributions of this work can be summarized as follows: We present a novel method for online drift-aware input time series variables selection using relevance and redundancy analysis; The drift detection mechanism is devised to operate on both spatial and temporal dimensions; We fully automate the choice of relevance and redundancy measures for MTS, as well as forecasting model selection using meta-learning; We provide a comparative empirical study with SoA methods, and discuss their implications in terms of predictive performance and scalability.

## 2 Literature Review

In contrast to univariate time series forecasting, i.e. the forecast of a single time series, where several methods for online adaptive single model selection [19] or ensemble learning [20, 21] have been proposed, most of existing methods for MTS forecasting are devised to operate in a static manner [26, 27, 9]. In other words, the models in these methods are learned offline using a collection of historical MTS data, their parameters are optimized using these datasets and stored to be used at test time to make the predictions. In addition, most of these methods are either application specific [9] or model specific, i.e. use an arbitrary selected

<sup>1</sup> [https://www.dropbox.com/sh/z2g0us0nti3nqzg/AAAJ6\\_6JcGZHN\\_y10q8XDYa\\_a?dl=0](https://www.dropbox.com/sh/z2g0us0nti3nqzg/AAAJ6_6JcGZHN_y10q8XDYa_a?dl=0)

machine learning model family [26]. The most widely used models are VAR [26, 20] or DNNs [27, 22]. In [9], MTS for energy forecasting in smart buildings is transformed into a standard regression task using time series embedding, and then, different types of feature selection methods for regression tasks are applied. The features are extracted offline once and kept static at test time.

More recently, some works have exploited the success of some DNNs architectures in computer vision-related applications and successfully transferred and adapted them to MTS forecasting by treating the temporal and the spatial dimensions in MTS as the 2d-dimension in images. Some of other works focused on introducing some improvements or adaptations over existing DNNs to cope with the characteristics of MTS [27, 22]. In [22], authors argued that the random weights initialization in Recurrent Neural Networks (RNNs), disallows the neurons from learning the latent features of the correlated variables of the MTS. Therefore, they suggest using a pre-trained LSTM combined with a stacked auto-encoder to replace the random weight initialization strategy adopted in deep RNNs. In [27], Graph Neural Networks (GNNs) are adapted to MTS forecasting by adding a mix-hop propagation layer and a dilated inception layer to capture the spatial and temporal dependencies within the MTS. This is done to make GNN capable of handling relational dependencies that are not known in advance like in the case of MTS. Even though dependencies between the variables of the MTS may change significantly over time, most of the aforementioned works do not consider a time-dependent selection of the input time series variables to the MTS forecasting model. The choice of the model is most often arbitrary or transferred from another domain like computer vision or regression. In addition, once the model is chosen, its corresponding parameters are kept fixed. It is important to note that there exist methods for model adaptation to data changes and model performance, more particularly to concept drift in the context of streaming data classification [14, 15] and univariate time series forecasting [21]. These methods can be grouped into two main families, namely blind adaptation and informed adaptation. In blind adaptation, the model is retrained either at each time instant with each upcoming observation or over a fixed period in time without any consideration of possible data or model performance changes. However, this family of methods is known to be time-intensive, resource-consuming, and unpractical for online forecasting [21, 19]. Informed adaptation methods use some statistical information about the data or model performance to inform the model about the occurrence of concept drift and, if necessary, trigger input data update using adaptive time-windowing approaches and input re-selection [14, 15, 21] and subsequently, model retraining [20] or new model selection [19]. In this work, we propose an informed adaptation for MTS forecasting. This is done by monitoring the changes in spatio-temporal dependencies in the MTS and the model performance over time. Since the results achieved in various time series tasks such as clustering and classification depend to a large extent on the used measures for evaluating time series dependencies/similarities [16], we suggest automating the choice of the adequate dependencies measures as well

as the selection of the adequate model for a particular application by means of meta-learning.

### 3 Methodology

In this section, we present our framework and its main components. For given MTS data, the input time series variables to be used for forecasting are determined in a timely manner by computing their *relevance* to the target time series variable. Once the most relevant variables are identified, *redundancy* analysis through time series clustering is carried out to remove redundant variables. The choice of adequate time series measures for *relevance* and *redundancy* is determined beforehand by the meta-learning component that decides as well which model to be used for particular MTS data. Both input time series variables and model updates are triggered once a concept drift in the spatio-temporal dependencies among these variables or/and model performance is detected. Basically, either new variables are selected or time windows are adjusted to update the time series variables with recent observations. This depends on the nature of detected concept drift, i.e. whether it is on variables dependencies or model performance or on both.

#### 3.1 Preliminaries

A time series variable  $X^i$  with  $i \in \mathbb{N}$ , is a temporal sequence of values, where  $X_{1:t}^i = \{x_1^i, x_2^i, \dots, x_t^i\}$  denotes the sequence of  $X^i$  recorded until time  $t$  and  $x_j^i$  is the value of  $X^i$  at a time instant  $j$ . A MTS  $\mathbf{X}$  consists of multiple time series variables, i.e.  $\mathbf{X} = \{X^1, X^2, \dots, X^N\}$ , that are interdependent. The variables are assumed in this work to be recorded simultaneously with the same frequency. The MTS  $\mathbf{X}_{1:t}$  recorded until a time instant  $t$  can be formally described as  $N \times t$ -dimensional matrix, with  $\mathbf{X}_j = \{x_j^1, x_j^2, \dots, x_j^N\}$  which represent the *spatial* dimension of  $\mathbf{X}$  for a fixed time instant  $j$  and  $X_{1:t}^i$  represents the evolution of  $\mathbf{X}$  over the *temporal* dimension across the variable  $i$ .

Given a target time series variable  $X^r$ , the goal of online input variable selection is to determine which time series variables  $X^i, i \in [1, N] \setminus \{r\}$  should be fed into the forecasting model at time  $t$  to forecast the next value at time  $t+1$ . It is important to note that in MTS, each time series variable can play the role of the target variable and be predicted using the remaining variables, as it may be that only one or some variables need to be predicted. This is application dependent. However, the reasoning applied to one target variable can be generalized to all the remaining variables. We denote by  $X_{t_s:t_e}^i$  the subsequence of  $X^i$  starting at time instant  $t_s$  and ending at time instant  $t_e$ . We divide the MTS  $\mathbf{X}$  into  $\mathbf{X}^{train} = \{X_{1:t-\omega}^1, X_{1:t-\omega}^2, \dots, X_{1:t-\omega}^N\}$  and  $\mathbf{X}_\omega^{val} = \{X_{t-\omega+1:t}^1, X_{t-\omega+1:t}^2, \dots, X_{t-\omega+1:t}^N\}$ , with  $\omega$  a provided window size.  $\mathbf{X}_\omega^{train}$  is used for training the forecasting model and  $\mathbf{X}_\omega^{val}$  is used to compute the *relevance* and *redundancy* measures, since both input and target time series variables are required to be known.

### 3.2 Forecasting Models Learning

Standard approaches for addressing MTS forecasting tasks include traditional techniques for MTS analysis, such as the popular Vector Autoregressive VAR family of methods [26], or ARIMAX [3] which is the extension of Autoregressive Integrated Moving Average model (ARIMA) to MTS where some input time series variables are provided as exogenous variables to forecast the dependent variable, i.e. target variable. These models take as an input multiple time series sequences  $\mathbf{X}_{1:t}$ . In addition, regression models can be employed in the context of MTS forecasting by using a time-delayed embedding that maps a set of observations from the target time series variable  $X^r \in \mathbf{X}$  to a  $l \times N$ -dimensional feature space corresponding to the  $l$  past lagged values of each observation in each time series variable in  $\mathbf{X}$ . Each observation is composed of a feature vector  $z_i \in \mathbb{Z} \subset \mathbb{R}^{l \times N}$ , which denotes the previous  $l$  values of each variable, and a target vector  $x_i \in \mathbb{X} \subset \mathbb{R}$ , which represents the value we want to predict. The objective is to construct a model  $f : \mathbb{Z} \rightarrow \mathbb{X}$ , where  $f$  denotes the regression function. In this work, we aim to select an adequate model given the characteristics of MTS data in question. This is done by the meta-learning components in Section 3.5. Therefore, we consider a pool of candidate forecasting models  $\mathbb{P}$  which is designed to contain a set of various and *heterogeneous* models, such as VAR, Gaussian processes, support vector regression, and DNNs. The candidate models are trained on  $\mathbf{X}_\omega^{train}$  using the same number  $l$  of lagged values for each variable in the MTS as input to model the following value in the time series.

### 3.3 Adaptive Input Time Series Variables Selection

Given a target time series  $X^r \in \mathbf{X}$ , in order to forecast its value at a future time instant  $t + h, h \geq 1$  (for simplicity of notation, we assume  $h = 1$ ), the selection of the time series variables  $X^i, i \in [1, N] \setminus \{r\}$  whose  $l$ -lagged values will be used as input for the forecasting model in addition to the  $l$ -lagged values of target time series  $X^r$ , has to be determined in a timely-manner at  $t$ . The selection is decided by measuring how much each of  $X^i, \forall i \in [1, N] \setminus \{r\}$  is relevant to  $X^r$  and whether  $X^i$  is redundant in the presence of the other variables.

**Relevance** The relevance of each  $X^i, \forall i \in [1, N] \setminus \{r\}$  to  $X^r$  is measured by computing the similarity between them on  $T_\omega^{val} = [t - \omega + 1, t]$ , denoted  $s_t^{i,r} = sim(X_{t-\omega+1:t}^i, X_{t-\omega+1:t}^r)$ . The time series variables  $X^i, \forall i \in [1, N] \setminus \{r\}$  are sorted according to their  $s_t^{i,r}$  and the top- $n$  most similar variables to  $X^r$  are selected. There is no single universal similarity measure between time series that is valid for every application. The choice of the adequate similarity measure is done by considering the characteristics of the MTS at question.

**Redundancy** The top- $n$  selected input time series variables may include some redundant variables that would lead to increasing the dimensionality of the MTS forecasting task without contributing to the model’s accuracy. Relying on the

computed similarity measures is not sufficient since they are measured over a time window of observations  $T_\omega^{val}$ . For instance, two candidate variables can have the same level of similarity to the target variables while being effectively similar to it on two distinct time intervals included within  $T_\omega^{val}$ . Therefore, we suggest removing redundancies by clustering the top- $n$  variables and selecting only one-time series representative per cluster. To compute clusters for time series, several techniques are proposed in the literature which can be classified based on the way they treat the data and how the underlying grouping is performed [1]. One classification depends on whether the whole series, a subsequence, or individual time points are to be clustered. In our case, we cluster the subsequences  $\{X_{t-\omega+1:t}^1, X_{t-\omega+1:t}^2, \dots, X_{t-\omega+1:t}^N\}$ . On the other hand, the clustering itself may be shape-based, feature-based, or model-based. The choice of time-series representation and the clustering algorithm has a big impact on performance with respect to cluster quality and execution time [23]. Again, no single clustering method is universally valid and the success of the method depends on the characteristics of the time series data [1]. Denote with the  $c_t^{i,j}$  the clustering measure used for computing the distance between the two sequences  $X_{t-\omega+1:t}^i$  and  $X_{t-\omega+1:t}^j$ , with  $i, j \in TOP_n$  and  $Top_n$  denotes the subset of selected input time series variables, i.e.  $|TOP_n| = top_n$ . The choice of the clustering algorithm together with the corresponding distance measure is decided by the meta-learning component. Further details are provided in Section 3.5.

**Drift-aware Variables Selection Adaptation** Both relevance and redundancies are monitored continuously over time. For relevance, with each upcoming data observation at  $t+h, h \geq 1$ , we slide  $T_\omega^{val}$  by one step, i.e. to include the observation at  $t+h$ , and we measure  $s_{t+h}^{i,r}, \forall i \in [1, N] \setminus \{r\}$ . Then, we computed:  $s_{t+h}^{min} = \min_{i \in [1, N] \setminus \{r\}} s_{t+h}^{i,r}$  in order to determine the distance between the target sequence and the most dissimilar sequence within the  $N-1$  input variables. Then, we compare it to the initial calculated distance  $s_{t_i}^{min}$ . In our case,  $t_i = t$  indicates the start of the online forecasting stage. The distance is treated as time series where  $s_{t+h}^{min}$  is its value at time  $t+h$ .

**Definition 1 (Weak stationary Similarity).** *The similarity structure between a set of input time series variables and a target time series is said to be weakly stationary if the true mean of  $\Delta^s$  is 0, with:  $\Delta_{t+h}^s = |s_{t+h}^{min} - s_{t_i}^{min}|$*

Following this definition, we can assume that the distance between the target time series sequence and the most dissimilar input sequence 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 an assumption using the well-known Hoeffding Bound, which states that after  $\omega$  independent observations of a real-value random variable with range  $R$ , its true mean has not diverged if the sample mean is contained within  $\pm\zeta$ :  $\zeta = \sqrt{\frac{R^2 \ln(1/\mu)}{2\omega}}$  with a probability of  $1 - \mu$  (a user-defined hyperparameter). Once the condition of the *weak stationary similarity* presented in Definition 1 is violated at  $t_d$ , a drift is assumed to take place at  $t_{d_s}$ . A relevance re-computation

is then triggered. A re-clustering is also performed, the selection of the variables is updated and the reference diameter  $s_{t_i}^{min}$  is reset by setting  $t_i = t_{d_s}$ . This drift type is denoted **Drift Type I**.

Similarly for the redundancy, we monitor continuously the distance measure used for clustering  $c_{t+h}^{i,j}, \forall X^i, X^j \in TOP_n$ , which results in the similarity matrix  $\mathcal{C}_{t+h} = (c_{t+h}^{i,j})_{1 \leq i, j \leq top_n} \in \mathbb{R}^{top_n \times top_n}$  and we place all the elements of  $\mathcal{C}_{t+h}$  in a vector  $\varsigma_{t+h}$ , where  $\varsigma_{j,t+h} \geq \varsigma_{j-1,t+h}, \forall j \in \{1, \dots, top_n^2\}$ . Let  $\varsigma_{t_i}$  denote the value of  $\varsigma$  at the initial instant  $t_i = t$  of the generation of  $\mathcal{C}$ . We monitor the deviation  $\Delta_{t+h}^{\varsigma} = |\varsigma_{t+h} - \varsigma_{t_i}|$  similarly to  $\Delta_{t+h}^s$ . We test the occurrence of concept drift within the clusters following the same condition defined in Definition 1. If a concept drift is detected at  $t_{d_c}$ , both relevance and redundancies re-computation are triggered and a re-selection of input variables is performed. We reset then  $\varsigma_{t_i} = \varsigma_{t_{d_c}}$ . This drift type is denoted **Drift Type II**.

### 3.4 Forecasting Models Adaptation

The increase in the forecasting error may indicate a possible change in the relationship between the input variables and the target time series or outdated model parameters due to outdated time series observations that were used for training. Therefore, necessary measures such as input variables re-selection and/or model re-training with recently acquired data have to be taken. To do so, the forecasting error  $\epsilon$  is estimated using the Root Mean Square Error (RMSE) and is monitored over the sliding window of the recent observations  $T_{\omega}^{val}$ . The error can be viewed as a time series, and at  $t+h$   $\epsilon_{t+h}^{\omega} = \frac{1}{\omega} \sum_{j=t+h-\omega}^{t+h-1} (x_j^r - \hat{x}_j^r)^2$ , with  $\hat{x}_j^r$  the predicted value of  $X^r$  at time  $j$ . Naturally, with time-evolving data, the model's error changes over time and may follow non-stationary concepts. Let  $\epsilon_{t_i}$  denote  $\epsilon$  value at the initial instant of its generation  $t_i = t$ . Since the forecasting error is directional, the drift-detection using the absolute value of the error deviation with the Hoeffding-bound can be misleading. Therefore, we suggest using the Page-Hinkley Test [20] to detect a significant increase in the forecasting error. We present the pseudo-code of the Page-Hinkley Test in the supplementary materials.  $\nu$  and  $\varrho$  are user-defined hyper-parameters, where  $\nu$  is the tolerable change in the estimated error and  $\varrho$  is a threshold. A larger  $\varrho$  avoids detecting false drift alarms, but can also lead to missing true drifts [8]. The error drift detection is denoted **Drift Type III**. An alert at time  $t_{d_c}$  declares the occurrence of **Drift Type III** and triggers the update of the input variables through new selection, i.e new relevance and redundancy re-computation and updates the current model with the new input and the recent observations. It also restarts the Page-Hinkley Test from the beginning. The model gets also updated with the update of the input triggered by **Drift Type I** or **Drift Type II**.

### 3.5 Online Automated MTS Forecasting

As discussed above, there are no single universal similarity measures for relevance and redundancies. Similarly, for the forecasting model, adequate model selection



has to be performed to cope with the characteristics of the MTS in question. Once the model is selected, the online adaptation scheme in our framework (See Section 3.4) takes care of the update of the model in an informed manner to the real-time changes in the data and the performance. To automate the choice of the measures and the model, we use meta-learning. Let  $\mathbb{S}$  and  $\mathbb{C}$  be the spaces of the relevance and redundancy measures, respectively. Denote with  $\mathbb{M}$  the space of the candidate models to solve the MTS forecasting task. Using a set of  $m$  MTS characteristics represented here by the so-called meta-features, the goal of the meta-task is to fit model  $f_{meta} : \mathbb{R}^m \rightarrow \mathbb{S} \times \mathbb{C} \times \mathbb{M}$  to predict the best combination of relevance and redundancies measures and forecasting model choice given a vector of  $m$  MTS meta-features as input.

**MTS Meta-features** Several works have been proposed for extracting Univariate Time Series (UTS) meta-features [25]. Therefore, most of the existing works that tackled the same task for MTS use the same features developed for UTS to extract meta-features from each time series variable in the MTS and concatenate them in one feature vector [9]. In this work, in addition to the transfer of the most often used meta-features in the context of univariate time series to the MTS domain, we propose to add MTS-specific meta-features. We additionally adapted the concept of *land-marking* developed for meta-tasks in classification and regression [12] to MTS data. The extracted meta-features can be grouped into three main families.

*UTS-specific features* For each time series variable in the MTS, we extract different time series-specific features that can be grouped into three families, including descriptive statistics, frequency domain, and auto-correlation features [11]. The list of these features includes then trend, skewness of series, turning points, kurtosis of series, step changes, length of series, non-linearity measure, the standard deviation of de-trended series, power spectrum: maximal value, no. of peaks not lower than 60% of the max, auto- and partial correlations at lags one and two, seasonality. Since the number of variables in the MTS can be very big, we compute the mean and the standard deviation of each extracted feature over the different variables from a subset.

*MTS-specific features* We suggest investigating the relationships/dependence among the MTS variables. To do so, we compute several similarity measures [16], including Pearson Correlation, Euclidean distance, Dynamic time warping distance, Mahalanobis distance, Amplitude and Phase differences of the Fourier Transform (FT), and Shape similarity based on derived FT amplitude and phase differences, between each pair of variables. These similarities computations result in similarity matrices for each measure. Instead of concatenating all the coefficients of all the matrices in one feature vector and increasing the meta-task input dimensionality, we suggest computing diversity in similarity/dependence along with all the variables pairs for each similarity matrix. Denote with  $\mathcal{S} \in \mathbb{R}^{N \times N}$  the resulting similarity matrix of a given similarity/distance  $s$  between all the

$N$  MTS variables. We define the diversity as (note the similarity values are normalized between -1 and 1) :  $div(\mathcal{S}) = 1 - \frac{1}{\sum_{1 \leq i \neq j, \leq N} s(X^i, X^j)}$

*Landmarking-based features* This type of meta-features are designed to describe the performance of some learning algorithms, called *landmarkers*, in various learning contexts on the same data. *Landmarkers* are machine learning models that are computationally relatively cheap either in training or testing compared to other models. So far, all the proposed *landmarkers* and corresponding meta-features have been proposed for classical meta-learning applications to classification problems and one work has added the extension of this concept to regression [13], whereas we focus on *landmarkers* integration for MTS forecasting. In regression, the process starts by creating one landmarking model over the entire training set. A small artificial neighborhood for each training example is created using Gaussian noise. Then descriptive statistics of the models' output, mean, stdev., 1st/3rd quantile, are extracted. In our case, we use, LASSO, 1NN, MARS and CART, as *landmarkers* [13] and train them on  $\mathbf{X}_\omega^{train}$ . We can distinguish three types of Landmarking features:

- *Global landmarking*: We evaluate each model on each time  $\mathbf{X}_\omega^{val}$  and we extract the descriptive statistics of the models' output.
- *Performance-based local landmarking*: we split  $\mathbf{X}_\omega^{val}$  into equally-sized non-overlapping time windows of size  $n_\omega$ . We evaluate each model on each time window and we extract for each window the descriptive statistics of the models' output.
- *Model-based local landmarking*: This type of local landmarking is designed to characterize the *landmarkers* within a particular time series region, in our case each time window of size  $n_\omega$ . To do so, we extract the knowledge that the *landmarkers* have learned about each window. In addition to the prediction of each *landmarker* on each window, we compute the depth of the leaf which makes the prediction and the number of examples in that leaf and variance for each window for CART, the average over each window of the width and mass of the interval in which each time value falls, and the average over each window of absolute distance to the nearest neighbor for 1NN.

## 4 Experiments

We present the experiments carried out to validate OAMTS and to answer these research questions: **Q1**: How does OAMTS perform compared to the SoA and existing online methods for MTS forecasting?; **Q2**: To which extent is it necessary to automate the choice of adequate relevance and redundancies measures, as well as the forecasting model choice? **Q3**: What is the importance of each component, namely relevance and redundancy, in the input time series variables selection on the performance? **Q4**: What is the benefit of each drift type detection for the performance of OAMTS? **Q5**: How scalable is OAMTS in terms of computational resources compared to the most competitive online model selection methods?

and what is the computational advantage of **drift-aware** adaptation of the framework?

#### 4.1 Experimental Setup

The methods used in the experiments were evaluated using the root mean squared error (RMSE). We collected a total of 166 MTS from various real-world applications. 100 MTS are exclusively used for the meta-learning task, while the remaining 66 MTS are used for testing the meta-model which recommends which relevance and redundancy measures and forecasting model from the pool of candidate models that we have devised, to use. Following the recommendation of the meta-model, these 66 MTS are used to validate the online forecasting performance of OAMTS. Each of the 66 MTS was split using 50% for training ( $X_{\omega}^{train}$ ), and 25% for validation ( $X_{\omega}^{val}$ ) and 25% for testing. Note that in each MTS, we have chosen one variable as the target one depending on the application and the remaining variables as different input variables. However, for some applications like taxi demand forecasting, all the variables can play the role of the target one and change the role between variables. A full list of the used datasets, together with a description, is given in the code repository <sup>2</sup> and in the supplementary materials.

**Candidate models set-up** We construct the pool  $\mathbb{P}$  of candidate models. We mentioned earlier that there is no single method for forecasting that outperforms all the other methods on every time series. Hence, we incorporate and test different families of models. Traditional time series forecasting models like **VAR** [26] is included. Regression models are also included in  $\mathbb{P}$  and are applied after using MTS embedding of dimension  $N \times l$ . These models include Gradient Boosting Machines **GBM** [5], Support Vector Regression **SVR** [4], Random Forest **RF** [2], Projection Pursuit Regression **PPR** [6], MARS **MARS** [7], and Partial Least Squares Regression **PLS** [17]. Neural networks based models that are designed for time series forecasting task are introduced to  $\mathbb{P}$  such as Multi-Layer Perceptron **MLP** [10], Bidirectional LSTM **bi-LSTM** [24]. More recently, **CNN-LSTM** [28] and Convolutional LSTM **Conv-LSTM** [28] are suggested to solve MTS forecasting tasks. Using different parameter settings for each family, we generate a pool of 20 candidate models.

**Meta-learning task set-up** The list of similarity measures considered to measure the variables relevance includes Pearson Correlation, Spearman correlation, Euclidean distance, Dynamic time warping distance, Manhattan distance, and Fourier-based distance. A detailed description of each measure can be found in [16] (Table1). For redundancies, we have chosen  $K$ -means [18] as the clustering algorithm with distance measure either Euclidean distance or Dynamic time warping distance. For the models, we consider the selection from the pool  $\mathbb{P}$ . Note for the meta-data labelling, we consider all the possible combinations of relevance, redundancy measures and model type and we evaluate

<sup>2</sup> [https://www.dropbox.com/sh/z2g0us0nti3nqzg/AAAJ6\\_6JcGZHN\\_y10q8XDYa\\_a?dl=0](https://www.dropbox.com/sh/z2g0us0nti3nqzg/AAAJ6_6JcGZHN_y10q8XDYa_a?dl=0)

our framework performance on each MTS dataset in the meta set by splitting it into 80% for training the framework and 20% for testing. Even though, the meta-task is performed fully offline (only meta-model predictions are output online), this annotation is very resource-consuming because of the big number of combinations. That is why we restrained the size of the metadata to 100 MTS. However, we aim to enlarge this data in the future. There are different options on how to tackle the meta-learning task. One possible option would be to encode all the combinations of relevance, redundancy measures, and model type which would lead to a high number of classes compared to the size of the meta-data. Another option is to consider it as a multi-label classification task. However, a classifier’s performance on different labels can vary significantly. Therefore, we have chosen to split the task into three learning tasks. The first one is for relevance measure prediction and is a multi-class classification task solved with SVM [4]. The second task is for redundancy measure prediction and is a binary classification task solved with SVM [4]. The third task is for model selection and is a multi-class classification task solved with RF [2]. The choice of the learning algorithm is decided using a cross-validation evaluation of the accuracy on the meta-data.

**OAMTS set-up:** **OAMTS** has also a number of hyper-parameters that are summarized in Table 1 in the supplementary materials. We compare **OAMTS** against the following approaches which include SoA methods for MTS forecasting. Some of them operate in an online fashion.

**SoA Forecasting Models:** **ARIMAX**[3]: Auto-Regressive Moving Average model with exogenous variables, **LSTM**[19]: Long Short Term Memory Network which has shown better performance than the remaining neural networks such as MLP and CNN-LSTM and comparable performance with bi-LSTM, **VAR**[26]: Traditional Vector Autoregressive model. Its order is tuned using Akaike Information Criterion (AIC) using the **R-package** ‘vars’, **Drift-aware VAR** [20] A recent framework that selects the relevant variables using Pearson-Correlation for the VAR model and update them following concept-drift detection. It uses also L1-regularization to prevent over-fitting. However, redundancies are not removed.

**OAMTS Variants:** **OAMTS-Ran:** The variant of OAMTS that is computed using a random selection of Relevance and Redundancies measures and model, **OAMTS-VAR:** The variant of OAMTS that uses VAR as the forecasting model instead of the automatic model selection. Relevance and Redundancies are selected by the meta-model, **OAMTS-Rel:** The variant of OAMTS that performs adaptive input selection by considering only the relevance, **OAMTS-Red:** The variant of OAMTS that performs adaptive input selection by considering only the redundancy, **OAMTS-DI-II:** The variant of OAMTS that performs model adaptation following concept drift in the input structure (**Drift type I** and **Drift type II**), **OAMTS-DIII:** The variant of OAMTS that performs model adaptation following in the changes in the error (**Drift type III**), **OAMTS-Per:** The variant of OAMTS that performs model adaptation periodically without any consideration of concept drift occurrence, with each upcoming 10% data points, **OAMTS-BG:** The variant of OAMTS where we assume we know the

background truth of which Relevance and Redundancies measures to use and which model to select. This is done by evaluating all the possible combinations on the test set. This variant is used as a reference model to know how well the meta-learning component performs.

## 4.2 Results

Table 1 presents the average ranks and their deviation for all methods. For the paired comparison, we compare our method OAMTS 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 (significance level 0.05). In the results in Table 1, OAMTS outperforms

Table 1: Comparison of OAMTS to different SoA for 66 time series. The rank column presents the average rank and its standard deviation across different time series. A rank of 1 means the model was the best performing on all time series. We report only significant wins and losses of OAMTS against remaining methods.

Method	Our Method		
	Wins	Losses	Avg.rank
VAR	40	0	7.7±0.9
ARIMAX	20	20	3.0±2.2
LSTM	40	0	7.8±1.4
Drift-aware VAR	40	0	6.7±0.9
OAMTS-VAR	40	0	7.6±0.7
OAMTS-Ran	40	0	5.0±0.9
OAMTS-Rel	40	0	5.9±1.3
OAMTS-Red	40	0	6.3±0.6
OAMTS-Per	39	1	4.3±0.8
OAMTS-DI-II	30	10	3.2±1.2
OAMTS-DIII	7	33	2.9±0.7
<b>OAMTS</b>	-	-	<b>2.2±0.5</b>
<b>OAMTS-BG</b>	-	-	<b>1.9±0.6</b>

the baseline methods in terms of wins/loses in pairwise comparison. The online MTS forecasting methods, e.g., Drift-aware VAR [20] and OAMTS-VAR show inferior performance compared to OAMTS. VAR and LSTM, SoA methods for forecasting, are considerably worse in average rank compared to OAMTS. The most competitive SoA approach to OAMTS is ARIMAX. Nevertheless, it has a higher average rank and a lower performance than our method. VAR is considered to be the most widely used method of MTS forecasting but it can be seen from OAMTS-VAR that it is not always the best model choice. This is also confirmed by the Drift-aware VAR performance. It can also be seen that none of OAMTS-Rel and OAMTS-Red is able on its own to reach the performance of OAMTS

which shows the importance of both relevance and redundancies consideration in the input selection. These results address the research questions **Q1-Q2**.

Table 2 presents some examples where we show the ground truth of which are the best relevance and redundancies measures, as well as the model choice for some data sets. It is clear from Table 2 that there is no one single best relevance

Table 2: Ground truth of the best model and relevance/redundancy measures for some datasets.

Dataset	Model	Similarity measure	Clustering method
<b>Taxi-1</b>	PLS	Pearson correlation	DTW
<b>Taxi-2</b>	MARS	Euclidean distance	DTW
<b>Taxi-3</b>	PLS	Spearman correlation	DTW
<b>Chengdu-city-3</b>	MARS	Spearman correlation	Euclidean

and redundancies measures, as well as one optimal model choice, even for MTS data sets extracted from the same data source like Taxi1,2,3 that are extracted from NYC Trip Record Data (Yellow taxi 2021). This justifies the necessity of automating these choices. Random choices would lead to considerably worse performance which is reflected in the performance of OAMTS-Ran in Table 1. In addition, comparing OAMTS to OAMTS-BG, we can see a slight difference in the ranks in favor of course of OAMTS-BG but it highlights the usefulness of the meta-learning component in our framework for automating all the choices. These results address the research question **Q3**.

From Table 1, we can also see that none of the drift adaptation methods is able on its own to perform as well as OAMTS which deploys the three drift types to monitor changes in the input dependence structure as well as the model performance. In addition, OAMTS which relies on the informed adaption of the framework using concept drift detection is better than OAMTS-Per. This can be explained by the fact that unnecessary updates are not always beneficial. This answers the research question **Q4**.

In the next experiment, we compare the runtime of OAMTS and its variants against some SoA methods in Table 3.

All the reported runtimes concern only the online predictions and any operation computed offline is not taken into account. The results demonstrate that OAMTS has lower runtime than OAMTS-Per. This is due to using drift detection to update only when necessary. This results in faster predictions and less computational requirements. The high deviation of the runtime of OAMTS is due to the different numbers of drifts per time series. This answers question **Q5**.

### 4.3 Discussion and future work

The empirical results indicate that OAMTS has performance advantages compared to popular MTS forecasting methods. We show that our method, for adaptively

Table 3: Empirical runtime comparison between different methods in Seconds.

Method	OAMTS	OAMTS-Per	LSTM
Avg. Runtime	34.26	72.12	150.09
±	94.51	35.29	29.26

selecting input MTS variables and performing the model update, is able to gain excellent and reliable empirical performance in our setting. The informed adaptation following concept drift detection makes our method in addition to better predictive performance, computationally cheaper than blind adaptation methods like periodic ones. In future work, we plan to enhance further the meta-learning components by adding more datasets and annotating them, and establishing a direct mapping to the best combination of measures and model choice as target label as we assume that there is a link in addition to the MTS characteristics that we tried to cover from different perspectives, between relevance and redundancies measures and the chosen forecasting model. This investigation will make the scope of our future work. In addition, we’ve thought about adding more time series clustering algorithms so that we change the mapping to the clustering algorithm directly instead of the relevance measure. We may also think about enlarging the pool  $\mathbb{P}$ .

## 5 Concluding Remarks

This paper introduces OAMTS: a novel, practically useful online adaptive framework for multivariate time series forecasting. OAMTS uses adaptive input selection by investigating relevance and redundancies. Both input variables and learning models are updated in an informed manner following different types of concept drift detection. The choice of the relevance and redundancies measure, as well as the model, is automated using meta-learning. An exhaustive empirical evaluation, including several real-world datasets and multiple comparison algorithms, showed the advantages of OAMTS in terms of performance and scalability.

## References

1. Aghabozorgi, S., Seyed Shirshorshidi, A., Ying Wah, T.: Time-series clustering-a decade review. *Information Systems* **53**, 16–38 (2015). <https://doi.org/https://doi.org/10.1016/j.is.2015.04.007>, <https://www.sciencedirect.com/science/article/pii/S0306437915000733>
2. Breiman, L.: Bagging predictors. *Machine learning* **24**(2), 123–140 (1996)
3. Dissanayake, B., Hemachandra, O., Lakshitha, N., Haputhanthri, D., Wijayasiri, A.: A comparison of arimax, var and lstm on multivariate short-term traffic volume forecasting. In: Conference of Open Innovations Association, FRUCT. pp. 564–570. No. 28, FRUCT Oy (2021)
4. 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)

5. Friedman, J.H.: Greedy function approximation: a gradient boosting machine. *Annals of statistics* pp. 1189–1232 (2001)
6. Friedman, J.H., Stuetzle, W.: Projection pursuit regression. *Journal of the American statistical Association* **76**(376), 817–823 (1981)
7. Friedman, J.H., et al.: Multivariate adaptive regression splines. *The annals of statistics* **19**(1), 1–67 (1991)
8. Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: A survey on concept drift adaptation. *ACM computing surveys (CSUR)* **46**(4), 1–37 (2014)
9. González-Vidal, A., Jiménez, F., Gómez-Skarmeta, A.F.: A methodology for energy multivariate time series forecasting in smart buildings based on feature selection. *Energy and Buildings* **196**, 71–82 (2019). <https://doi.org/https://doi.org/10.1016/j.enbuild.2019.05.021>, <https://www.sciencedirect.com/science/article/pii/S0378778818338775>
10. Goodfellow, I., Bengio, Y., Courville, A.: *Deep Learning*. MIT Press (2016), <http://www.deeplearningbook.org>
11. Hyndman, R.J., Wang, E., Laptev, N.: Large-scale unusual time series detection. In: 2015 IEEE international conference on data mining workshop (ICDMW). pp. 1616–1619. IEEE (2015)
12. Khiari, J., Moreira-Matias, L., Shaker, A., Ženko, B., Džeroski, S.: Metabags: Bagged meta-decision trees for regression. In: *Joint european conference on machine learning and knowledge discovery in databases*. pp. 637–652. Springer (2018)
13. Khiari, J., Moreira-Matias, L., Shaker, A., Ženko, B., Džeroski, S.: Metabags: Bagged meta-decision trees for regression. In: *Joint European Conference on Machine Learning & Knowledge Discovery in Databases*. pp. 637–652. Springer (2018)
14. Klinkenberg, R., Joachims, T.: Detecting concept drift with support vector machines. In: *ICML*. pp. 487–494 (2000)
15. Klinkenberg, R., Rüping, S.: Concept drift and the importance of examples. In: *Text mining—theoretical aspects and applications*. Citeseer (2002)
16. Lhermitte, S., Verbesselt, J., Verstraeten, W.W., Coppin, P.: A comparison of time series similarity measures for classification and change detection of ecosystem dynamics. *Remote sensing of environment* **115**(12), 3129–3152 (2011)
17. Mevik, B.H., Wehrens, R., Liland, K.H.: *pls: Partial Least Squares and Principal Component Regression* (2018), <https://CRAN.R-project.org/package=pls>
18. Priebe, F.: *Dynamic model selection for automated machine learning in time series* (2019)
19. Saadallah, A., Jakobs, M., Morik, K.: Explainable online deep neural network selection using adaptive saliency maps for time series forecasting. In: Oliver, N., Pérez-Cruz, F., Kramer, S., Read, J., Lozano, J.A. (eds.) *Machine Learning and Knowledge Discovery in Databases. Research Track*. pp. 404–420. Springer International Publishing, Cham (2021)
20. Saadallah, A., Moreira-Matias, L., Sousa, R., Khiari, J., Jenelius, E., Gama, J.: Bright-drift-aware demand predictions for taxi networks. *IEEE Transactions on Knowledge and Data Engineering* (2018)
21. Saadallah, A., Priebe, F., Morik, K.: A drift-based dynamic ensemble members selection using clustering for time series forecasting. In: *Joint European conference on machine learning and knowledge discovery in databases*. Springer (2019)
22. Sagheer, A., Kotb, M.: Unsupervised pre-training of a deep lstm-based stacked autoencoder for multivariate time series forecasting problems. *Scientific reports* **9**(1), 1–16 (2019)
23. Sardá-Espinosa, A.: Comparing time-series clustering algorithms in r using the dtwclust package. *R package vignette* **12**, 41 (2017)



24. Sun, Q., Jankovic, M.V., Bally, L., Mougiakakou, S.G.: Predicting blood glucose with an lstm and bi-lstm based deep neural network. In: 2018 14th Symposium on Neural Networks and Applications (NEUREL). pp. 1–5. IEEE (2018)
25. Talagala, T.S., Hyndman, R.J., Athanasopoulos, G., et al.: Meta-learning how to forecast time series. *Monash Econometrics and Business Statistics Working Papers* **6**(18), 16 (2018)
26. Tsay, R.S.: *Multivariate time series analysis: with R and financial applications*. John Wiley & Sons (2013)
27. Wu, Z., Pan, S., Long, G., Jiang, J., Chang, X., Zhang, C.: Connecting the dots: Multivariate time series forecasting with graph neural networks. In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. pp. 753–763 (2020)
28. Xingjian, S., Chen, Z., Wang, H., Yeung, D.Y., Wong, W.K., Woo, W.c.: Convolutional lstm network: A machine learning approach for precipitation nowcasting. In: *Advances in neural information processing systems*. pp. 802–810 (2015)