Pattern Recognition 134 (2022) 109132

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Data augmentation for univariate time series forecasting with neural networks

Artemios-Anargyros Semenoglou*, Evangelos Spiliotis, Vassilios Assimakopoulos

Forecasting and Strategy Unit, School of Electrical and Computer Engineering, National Technical University of Athens, Athens, Greece

ARTICLE INFO

Article history:
Received 1 June 2022
Revised 5 October 2022
Accepted 22 October 2022
Available online 28 October 2022

Keywords:
Time series
Forecasting
Data augmentation
Neural networks
M4 competition

ABSTRACT

Neural networks have been proven particularly accurate in univariate time series forecasting settings, requiring however a significant number of training samples to be effectively trained. In machine learning applications where available data are limited, data augmentation techniques have been successfully used to generate synthetic data that resemble and complement the original train set. Since the potential of data augmentation has been largely neglected in univariate time series forecasting, in this study we investigate nine data augmentation techniques, ranging from simple transformations and adjustments to sophisticated generative models and a novel upsampling approach. We empirically evaluate the impact of data augmentation on forecasting accuracy considering both shallow and deep feed-forward neural networks and time series data sets of different sizes from the M4 and the Tourism competitions. Our results suggest that certain data augmentation techniques that build on upsampling and time series combinations can improve forecasting performance, especially when deep networks are used. However, these improvements become less significant as the initial size of the train set increases.

1. Introduction

For many decades, simple statistical methods (e.g. exponential smoothing) have dominated the field of time series forecasting, both in the academia and the industry, offering relatively accurate, fast to compute, and interpretable forecasts [8]. As more computational resources and more sophisticated machine learning (ML) algorithms became available [35], forecasting methods evolved, allowing for more accurate, scalable, and generalizable approaches that soon received considerable attention from the forecasting community.

Despite the enthusiasm surrounding the ML methods, in the area of univariate time series forecasting their adoption has been relatively slow, while their superiority over existing approaches widely questioned [23]. This is because many empirical studies, including popular forecasting competitions [7], have been reporting mixed results about the accuracy of the ML methods when used to forecast at scale, i.e. in applications where numerous, diverse series are to be forecast. What seems to have contributed most to the discrepant results was the utilization of ML algorithms for the development of "local" models, trained in a series-by-series fashion. In contrast to "global" models that learn from multiple series simultaneously how to forecast the individual ones, local models focus on each series separately, thus ignoring data correlations and patterns shared among the series, while also depending on fewer training samples [14].

The introduction of "cross-learning" in the M4 forecasting competition [24] changed the landscape in batch time series forecasting, popularizing the use of global ML models and highlighting their learning abilities. Subsequent studies that compare local and global models on artificial [10] and real data [30] have further strengthened the case for cross-learning, while most state-of-the-art ML forecasting methods, such as the DeepAR [29] and the N-BEATS [26], are implemented using globally trained models. More recently, global modeling has also reported excellent results in transfer learning forecasting settings [39].

Although in principle cross-learning can result in better forecasts, accuracy improvements should not be taken for granted. This is due to two major factors potentially affecting the performance of global forecasting models: *data representativeness* and *data set size*. In order for a ML model to properly learn how a series should be extrapolated in time, it is necessary for the underlying patterns of the series to be represented in the set used for training the model. In contrast to statistical methods, typical ML models do not prescribe the data generating process of the series and make little assumptions, meaning that they are incapable of extrapolating particular types of trends or seasonal patterns, unless such patterns can be observed in the train set. As a result, global models are

* Corresponding author.
E-mail addresses: artemis@fsu.gr (A.-A. Semenoglou), spiliotis@fsu.gr (E. Spiliotis), vassim@fsu.gr (V. Assimakopoulos).

https://doi.org/10.1016/j.patcog.2022.109132
0031-3203/

generally expected to work better for relatively homogeneous data sets in terms of time series features compared to diverse ones [34]. Moreover, since ML models naturally put more emphasis on the most popular samples of data to minimize the forecast error during training, it becomes evident that less popular types of series will most likely be forecast less accurately compared to the more popular ones.

However, even if all series needed to be forecast are sufficiently represented in the train set, a limited number of training samples could still be an issue. Most ML methods, especially neural networks (NNs), involve numerous trainable parameters, thus requiring large enough data sets to be properly fitted. The deeper a NN becomes to allow for a more detailed pattern recognition search, the more the number of the trainable parameters increases. Inevitably, the implementation of complex, global ML models becomes challenging in practice, unless the data set available for training consists of long series, each contributing multiple training samples to the train set, or multiple series, collectively resulting in an adequately large train set.

To mitigate the impact of insufficient data representativeness and limited sizes of train sets, researchers in computer science have proposed various data augmentation techniques that can be used to artificially increase the number of samples originally available for training the ML models [21]. Indeed, such techniques can become particularly useful in practice if we consider that, with the exception of some forecasting applications that naturally involve numerous, homogeneous series (e.g. retail sales, energy, and financial forecasting), more often than not, forecasters deal with small data sets that consist of relatively short, heterogeneous series. Nevertheless, in the field of time series analysis, data augmentation has been mostly considered for time series classification and anomaly detection tasks. Limited work has been conducted to evaluate the impact of data augmentation techniques in univariate time series forecasting settings, with most of the studies focusing either on particular types of data augmentation techniques or forecasting methods [3]. Moreover, little is currently known about the effect that such techniques have on different sizes of train sets, especially when used to train NNs.

To shed more light on the above research areas, in the present study we empirically evaluate the impact of data augmentation on the accuracy of NN univariate forecasting models of different sizes. We examine the forecasting performances on data sets consisting of a significantly different number of series, from different domains and of different frequencies. To do so, we consider both simple and sophisticated data augmentation techniques found in the literature, but also introduce a new one, to be called “upsampling”. Overall, the contribution of our study is summarized as follows:

- We empirically evaluate the impact of data augmentation on NN univariate forecasting models considering nine different techniques to artificially increase the size of the data set originally available. This includes both simple data augmentation techniques that can be applied in an online fashion, i.e. while training the forecasting models, and sophisticated ones that require off-line data processing.
- We investigate the potential accuracy improvements of data augmentation with respect to the size of the original data set originally available for training using a set of 23,000 real time series (large-sized data set) and two subsets of it consisting of 2300 (medium-sized data set) and 230 (small-sized data set) series, respectively. Drawing from our findings, we make recommendations about when data augmentation becomes more relevant for improving forecasting performance. Furthermore, we evaluate all approaches on three additional sets of series, consisting of yearly, quarterly and monthly data from the industry sector, in order to further validate our conclusions.
- We examine the effect of data augmentation with respect to the size and the complexity of the neural networks utilized for forecasting. Two feed-forward neural network architectures are considered, the first consisting of three relatively small dense layers and the second consisting of seven larger dense layers.
- We propose a novel data augmentation technique that randomly upsamples parts of the original training samples. Effectively, the proposed technique slices the existing samples and then “stretches” them back to their original length using interpolation to fill the data points missing. As a result, local time series features and patterns that may have otherwise been overlooked are emphasized.

The rest of the paper is structured as follows: Section 2 provides a brief review of the work done in time series augmentation, discussing noticeable gaps in the literature. Section 3 describes existing data augmentation techniques and introduces the upsampling one. Section 4 presents the data sets used for the empirical evaluation of our study, the forecasting models and benchmarks considered, and the overall experimental setup. Section 5 presents and discusses the results of the study, while Section 6 concludes the paper.

2. Related work

The adoption of sophisticated ML models in many applications has naturally drawn the attention of researchers seeking for richer, more balanced, and more representative data sets to data augmentation. In the field of computer vision there are already plenty of publications on the topic, proposing the use of geometric transformations (flipping, rotating, cropping, and translation), photometric (color space) transformations, and artificial generation of samples, among others [32]. In natural language processing the work done is more limited, including however a variety of rule-based, example interpolation, and model-based techniques [20].

Following the developments in computer vision and natural language processing, data augmentation techniques have been also proposed to support time series analysis. In their survey, Iwana et al. [13] provide a useful taxonomy of such techniques that includes random transformations, pattern mixing, generative models, and decomposition-based techniques applied on the time, magnitude, and frequency domain of the series. Simple approaches such as flipping vertically or horizontally the series, adding random noise to the series, and combining the series with randomly generated curves (warping) have been also successfully used for anomaly detection [38] and classification [36]. In pattern mixing, the work of Forestier et al. [9] is indicative, examining different averaging schemes that combine series in order to create synthetic data.

Data augmentation techniques that rely on decomposing the original series to key components are also well-established. For instance, Kegel et al. [17] assumed that a deterministic model can be used to decompose the series into a level, trend, seasonality, and residual component. Consequently, they created synthetic series that resemble the original one by adjusting the components of level, trend, and seasonality and adding a stochastic component based on the residuals. Similarly, Bergmeir et al. [4] suggested that synthetic series can be created from a decomposed series by bootstrapping the residual component using moving block bootstrap (MBB).

A more sophisticated class of data augmentation approaches relies on developing models that generate new time series based on an input that represents the desired features of the synthetic series or a random seed. Such models provide greater control over the characteristics of the synthetic data and, as a result, they can be used for selective augmentation. Both statistical [28] and ma-

chine learning [40] generative methods have been introduced over the years with great success.

In short, a large number of different data augmentation techniques have been proposed and studied in the literature. A comprehensive review of the most popular approaches can be found in the work of Wen et al. [37]. However, most of the studies have focused on time series classification and anomaly detection tasks, largely ignoring the impact of data augmentation on univariate time series forecasting. As a result, a more in-depth investigation of data augmentation in forecasting is required to provide useful insights and indicate promising directions for future research.

3. Data augmentation methods

Since the literature involves several techniques that can potentially be used to augment time series data, it is challenging in practice to investigate all of them in a single study. Therefore, we focus on an indicative set of the techniques available that effectively cover all the major types of approaches identified by Iwana et al. [13]. For each of the nine techniques examined we provide a brief, comprehensive description and an illustrative visual example of their use in Fig. 1, explaining also their main advantages and how they were explicitly implemented in our experiments.

Note that, as it will be described in Section 4.1, the series to be augmented will be forecast using training samples that consist of 24 consecutive historical observations in total. The first 18 points are essentially the input (predictor variables) of the NN forecasting model, while the last 6 its output (target variables). To facilitate training, the input vector is scaled in a range of [0,1] using the min-max transformation, according to which the output vector is also scaled. This means that the values of the output vector will not necessarily fall within the [0,1] range. Respectively, after applying any data augmentation technique, the synthetic series will be re-scaled accordingly.

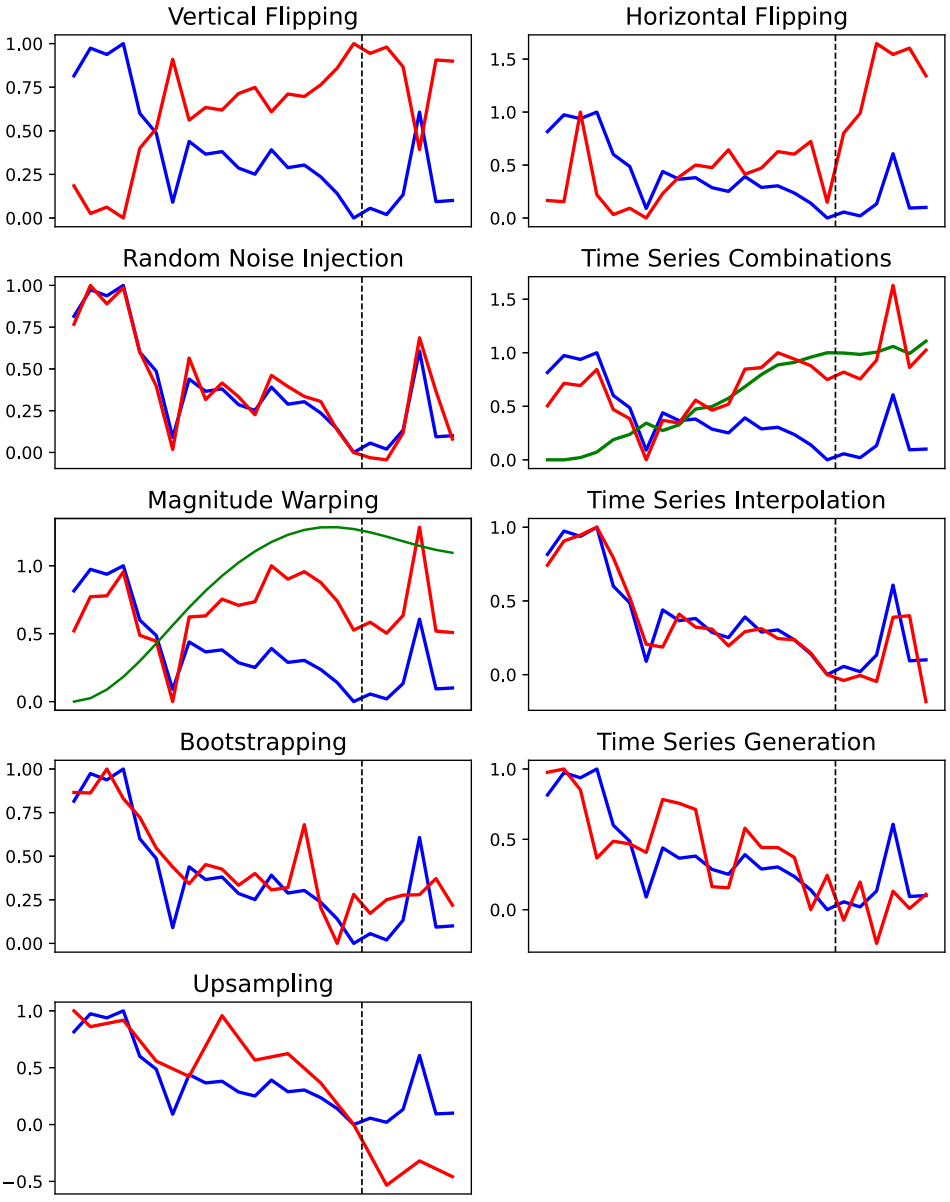

Fig. 1. Examples of time series augmentations performed by the techniques considered in this study. In each plot, the synthetic sample (red line) is created by employing the corresponding augmentation method on the original sample (blue line). The dashed line represents the barrier between the input and the output part of each sample. For the case of time series combinations and magnitude warping, the secondary original series and the randomly generated smooth curve are also included in the respective plot (green line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

3.1. Time series flipping

A popular data augmentation technique, especially in the field of computer vision, is based on rotating, or "flipping", the training samples along an axis. When flipping a series, most of its original features, such as the entropy and the autocorrelation, will remain the same. However, some other features, such as the trend, will change. As a result, the augmented series will share similar characteristics with the original one, which is a key advantage of the approach when interested in increasing the representativeness of particular series in the train set. Additionally, this data augmentation technique is easy to implement and fast to apply.

For the purposes of this study, two variants of the flipping technique were considered for augmenting the original time series, namely vertical and horizontal flipping. In the case of flipping the series vertically, the highest data point in the original sample becomes the lowest in the synthetic sample, and vice versa. The rest of the observations are adjusted accordingly. As a result, the original scale of the series is maintained. In the case of flipping the series horizontally, the order of the observations is simply reversed.

3.2. Random noise injection

Another commonly used augmentation technique is the introduction of random noise to the original data. In time series analysis settings, each data point of the series is either multiplied (multiplicative random noise) or added (additive random noise) to a random value. Typically, the random values are generated by assuming a normal or uniform distribution of predefined parameters.

There are two key points of advantage in using random noise injection for creating new training samples. First, in theory, the technique can create an infinite number of new samples, thus being capable to increase the overall size of the train set to any desired figure. Second, when properly tuned, random noise can improve the generalization of the forecasting model by introducing variance to the series but maintaining its fundamental patterns that are critical in forecasting. The latter property can become particularly useful when working with small data sets where deep NN models are more likely to overfit. Moreover, this technique is easy to implement and fast to compute.

In the present study, random noise values were produced by assuming a normal distribution with a mean $\mu = 0$ and a standard deviation $\sigma = 0.1$. The generated noise vector was added (additive noise) to the scaled observations of the original time series to create new samples.

3.3. Time series combinations

Combining the samples originally available in the train set is another straightforward and intuitive approach to augment time series data. Such combinations can be performed either through simple operators, such as the mean and the median, or more sophisticated aggregation schemes, as proposed for instance by Forestier et al. [9]. Moreover, combinations may involve two or multiple training samples. In general, considering combinations in pairs allows the creation of more artificial training samples that are also more diverse in terms of time series features, in contrast to combinations of multiple samples that, in extreme cases, will resemble the "average" sample of the original train set. Apart from being easy to implement and fast to compute, this technique has also the advantage of resulting in new samples that may either closely mimic the existing ones or differ from them to a noticeable extent, thus improving the overall generalization of the forecasting model.

For the sake of simplicity, in this study a linear combination of the original samples was considered. According to this approach, each synthetic sample is created by randomly selecting a pair of scaled training samples and, subsequently, combining them using the mean operator. Effectively, the new sample will be the mean of the two contributing samples.

3.4. Magnitude warping

Magnitude warping is a data augmentation technique based on warping the scale of a series by combining it with a randomly generated smooth curve. Depending on the shape and the scale of the curve, an additive or a multiplicative transformation of the series can be employed. Although magnitude warping changes the original series in a random fashion, this approach differs from that of the random noise injection in the sense that the former technique ensures that all changes follow a specific smooth sequence, as defined by the curve. Similarly, although magnitude warping builds on combinations, it differs from sample combining in the sense that the curve may not necessarily be representative of the originally available training samples. Therefore, this technique mixes to some extent the advantages of the two data augmentation approaches described above.

In this study, the case of an additive transformation was examined that uses a cubic spline with four knots. The corresponding curves were calculated by fitting the splines to data points, generated randomly using a normal distribution with a mean $\mu = 0$ and a standard deviation $\sigma = 0.2$.

3.5. Time series interpolation

Methods based on interpolation have been employed in different time series analysis tasks, both for filling missing values and augmenting the available data. Interpolation relies on actual observations for constructing a function that can be sampled in order to acquire new, artificial, observations. In its simplest form, a straight line can be derived from two already known data points and, then, be used for generating new points at any time interval between the original two points. Similarly, more complex functions that use several of the original observations can be fitted to provide more accurate approximations of the underlying data distribution. When interpolation is used for generating new time series, the synthetic samples are expected to closely mimic the long term patterns of the original data. In certain cases this can be advantageous, but, if more diversity is needed, methods based on interpolating the original series may fall short of the expectations.

In this study we implement such a data augmentation technique using the interpolation method proposed by Oh et al. [25]. This method was developed for time series classification but it can also be used in regression settings. As in the original paper, a cubic spline is fitted to the original series and then sampled at random intervals between the original observations, setting the data collection cycle's ratio $R = 10$.

3.6. Bootstrapping

This technique augments time series data by decomposing the original series into structural components and bootstrapping the remainder towards the creation of new series with the same underlying structure but different random components. Bergmeir et al. [4] proposed such an approach that first applies a Box–Cox transformation on the original series to stabilize the variance of the series and convert multiplicative patterns into additive ones, and then decomposes the transformed series into its components using Loess STL decomposition (trend, seasonal, and remainder) for seasonal data and Loess decomposition (trend and remainder) for non-seasonal data. The remainder component is bootstrapped using the MBB algorithm or the sieve bootstrap method towards the

creation of new vectors of remainders that follow the empirical distribution of the original remainder vector. The bootstrapped remainder vectors are then added to the other extracted components from the decomposition step to form new series. Finally, an inverse Box-Cox transformation is applied on the bootstrapped series to bring them back to the same scale as the original data.

Although bootstrapping shares some principles with random noise injection, in contrast to the latter, the former approach successfully retains the structural components of the original series, while introducing a noise component that originates from the examined series instead of being externally generated by a random process. As such, the main advantage of the bootstrapping technique is that it constructs samples that closely represent the original series, while also contributing variants of it that may be proven helpful in better generalizing the forecasting model [27]. Moreover, for each series available, bootstrapping can effectively create numerous new samples, thus facilitating the creation of large train sets. On the negative side, bootstrapping is computationally expensive compared to the techniques previously described.

In this study, bootstrapping was implemented following the suggestions of Bergmeir et al. [4] and using the MBB method for bootstrapping the remainder component.

3.7. Time series generation

By observing how the aforementioned techniques approach data augmentation, it becomes evident that all the artificial series to be created by these methods will mimic to some extent the structure of the original series. Although this is typically desirable, in some settings forecasters may be more interested in enriching the train set with types of series that are either highly underrepresented or completely absent. Moreover, forecasters may be interested in selectively augmenting a particular type of series instead of augmenting the complete train set in a global fashion. To allow for such a selective augmentation, researchers have introduced techniques that use models to generate new time series based on an input that summarizes the desired features of the artificial series. Such generative approaches may involve mixture autoregressive models [15], Gaussian tree models [6], convolutional NNs [19], and conditional generative adversarial networks [40], among others.

Given that time series generation can become complicated in terms of design, parameters, and computational time, in the present study we generated artificial series using the approach proposed by Petropoulos et al. [28] that allows for selective augmentation at a relatively low computational cost. Similar to the bootstrapping approach, Petropoulos et al. [28] suggest that each series can be decomposed into a trend, seasonality, and randomness component using the classical multiplicative decomposition method [24]. Then, the strength of each component can be measured using appropriate indicators. By applying this approach in an inverted fashion, i.e. by defining the desired strength of each component, generating components characterized by the defined characteristics, and aggregating the results, artificial series can be successfully constructed. To make sure that the artificial series will sufficiently represent the original series, the strength of their components was defined by randomly sampling values from continuous uniform distributions of minimum and maximum boundaries equal to the respective minimum and maximum strengths of the series included in the original train set.

3.8. Upsampling

In addition to the existing data augmentation techniques described earlier, we introduce a novel approach that re-samples the original samples of the train set to create new ones that closely resemble the originals but focus on particular parts of them with the objective to emphasize local variations of the series that may have otherwise been overlooked. The upsampling approach was inspired by techniques used in signal and image processing but was appropriately adapted to match the particular requirements of the examined forecasting task.

Given a training sample, representing a time series window that consists of W data point, the algorithm randomly selects $K < W$ consecutive points. Then, linear interpolation is used to fill the $W - K$ points missing so that the resulting sample has a length equal to W. Effectively, the upsampling technique randomly slices the existing samples and then stretches them back to their original length, serving as a "magnifying glass" that concentrates on subsequences of the original series and reveals local features and patterns. Other advantages of this approach is that it is easy and fast to implement, allowing also the construction of multiple new samples from each series. The value of W depends on the input and output size of the forecasting NN, the former being typically selected so that the forecasting accuracy is optimized, while the latter so that it matches the forecasting horizon. The value of K is calculated based on the number of new points that are interpolated. One or more artificial points can be inserted between the original observations, but, as more points are being added, the new samples will resemble the original series less effectively. Overall, W and K can be adjusted in order to best meet the requirements of the forecasting problem at hand.

In this study, W was set equal to 24 for the yearly series of the M4 competition, since the optimal input window for the forecasting NNs is equal to 18 observations (Section 4.2) and the forecasting horizon is equal to 6 observations (Section 4.1). Also, K was set equal to 13 in order for a single data point to be interpolated between each pair of original observations. As a result, 12 artificial points are spread among the 13 original ones. Finally, since the generated sequence contains 25 points, the first point is dropped in order for the new sample to have a length of $W = 24$. The same process was followed to augment the data sets of the Tourism competition. For the yearly series W was set equal to 16 and K equal to 9. For the quarterly series W was set equal to 32 and K equal to 17. For the monthly series W was set equal to 96 and K equal to 49. Note that, alternatively, this technique can be implemented by interpolating more data points between each pair of original observations or by dropping excess data points from the end of the generated sequence, with minimal changes in the performance of the forecasting models. More details on that are provided as supplementary material.

4. Experimental setup

This section provides information on the data sets considered for evaluating the performance of the examined data augmentation techniques, the NN forecasting models that were used for producing the forecasts, the forecasting performance measures utilized, and the benchmarks used as standards of comparison.

4.1. Time series data

To empirically evaluate the impact of data augmentation on forecasting accuracy, we use the data sets of the M4 [24] and the Tourism [2] forecasting competitions, which are well-known data sets widely used for testing and benchmarking new forecasting approaches. More specifically, from the M4 data set, we use the 23,000 yearly time series that effectively cover a variety of data domains (micro, macro, finance, demographics, and other). The Tourism data set provides a more focused case study, since the series included come from the tourism sector. In this study, we consider the entire competition's data set, i.e. 518 yearly, 427

quarterly and 366 monthly series. The use of such large, popular, and publicly available data sets is critical in order for our results to be reproducible and directly comparable to those of relevant published studies.

Since our study focuses on data augmentation techniques, it is important to take into account in our experiments the size of the data set used for training the forecasting models. It is logical to assume that in the case of a large collection of time series, similar to the 23,000 yearly series of the M4 competition, an adequate number of training samples will already be available for effectively training a NN, even without using augmentation. As a result, data augmentation may be less imperative for improving forecasting accuracy compared to cases where less time series are originally available. Therefore, we consider three different train sets of different sizes, all originating from the initial data set of the 23,000 series of the M4 competition: a large data set, to be noted D_{Large}, consisting of all 23,000 series, a medium-sized data set, to be noted D_{Medium}, consisting of 2300 series that were randomly selected from D_{Large}, and a small data set, to be noted D_{Small}, consisting of 230 series that were randomly selected from D_{Medium}. Note that the three data sets are used independently for training and evaluating the NN forecasting models (e.g. if a NN is trained using D_{Small}, it is evaluated in forecasting only the series contained in D_{Small}).

All experiments involving the D_{Large}, D_{Medium}, and D_{Small} sets of series follow the original setup of the M4 competition. Therefore, the forecasting horizon is set equal to 6 years and, subsequently, the last 6 observations of each series are being used only for the final evaluation (out-of-sample data) of the forecasts. The remaining observations (in-sample data) are being used for hyper-parameter optimization and training purposes. When the in-sample part of the series consists of sufficient data points, the sliding window strategy [33] is used to extract multiple training samples, each consisting of 24 observations in total (18 to be used as input to the NNs and 6 to be used as output). Table 1 summarizes the number of series included in each data set originating from the M4, as well as the maximum number of training samples that can be extracted per case.

When using the Tourism competition's data for comparing the forecasting performance of data augmentation techniques, we effectively consider three sets of series, based on their frequency (monthly, quarterly, yearly). Similar to the M4, the rules of the original competition were followed. This means that the forecasting horizon is set equal to 4 years, 8 quarters, and 24 months ahead in the cases of the yearly, quarterly, and monthly series, respectively. The in-sample data of the series are used for training the models, while the remaining out-of-sample data are only used for evaluation purposes. Finally, the sliding window strategy is used to extract multiple training samples. Table 1 summarizes the number of series included in each data set of the Tourism competition, as well as the maximum number of training samples that can be extracted per case.

Table 1
Number of time series and available time series windows in the D_{Large}, D_{Medium}, and D_{Small} data sets from the yearly series of the M4 competition, as well as in the Yearly, Quarterly, and Monthly data sets of the Tourism competition.

Data set		Time series	Training samples
M4	D_{Large}	23,000	235,460
Competition	D_{Medium}	2300	24,275
	D_{Small}	230	2544
Tourism	Yearly	518	3231
Competition	Quarterly	427	25,900
	Monthly	366	65,754

Note that all training and test instances are transformed before being introduced to the forecasting models. Data scaling is considered standard practice when NN forecasting models are employed [31], facilitating training and model generalization. This practice is particularly important for global models where samples from different series of different scales have to be processed by the same model. For this purpose, we apply the simple mix-max transformation for adjusting the scale of each training sample to a target range of $[0, 1]$. The inverse transformation is applied to the output of the models to obtain the final forecasts.

4.2. Neural network forecasting models

To evaluate the impact of the examined data augmentation techniques on NN univariate forecasting models, we consider an ensemble (median) of 30 shallow feed-forward networks with fully-connected layers, in a similar fashion to Semenoglou et al. [30]. This type of multilayer percepton (MLP) models are widely used for constructing global, auto-regressive forecasting methods since they are easy to implement, fast to train, and relatively accurate when compared to other, more advanced NN architectures. The NNs used in the ensemble were developed in Python 3.7.6 using Google's Tensorflow API v.2.4.0.

To accurately implement the examined forecasting models, a set of hyper-parameters related to the architecture and training process of the networks had to be optimized. To do so, we employed the Tree-of-Parzen-Estimators (TPE) algorithm for performing a search, as implemented in the *HyperOpt* library for Python [5]. The validation set used for the optimization involved the last training sample of each of the 23,000 series. The optimal values, as determined by the search, were then used for building and training all models involved in the ensemble, irrespective of the examined data set and the use of a data augmentation technique. Note that a single set of optimal values was used across all experiments due to the high computational cost of the hyper-parameters optimization process, but also to make sure that the results of the individual experiments would be comparable. The architecture of the base forecasting MLP is presented in the left part of Fig. 2. Also, Table 2 contains a summary of all the optimized hyper-parameters, the search range for each hyper-parameter, and the values identified as optimal by the TPE algorithm.

Furthermore, it important to study the effect of different augmentation techniques in relation to the complexity of the NNs used for forecasting the series. For that reason, we also develop and train ensembles of 30 deeper feed-forward networks with fully-connected layers. More specifically, each deep MLP consists of 7 dense layers with 256 nodes each, with more than 400,000 trainable parameters in total. To facilitate training in such a deep network, simple residual connections were also implemented. Finally, the parameters of the training process are similar to those of the shallow MLP. An overview of the deeper MLP topology is presented in the right part of Fig. 2.

4.3. Benchmarks

In order to assess the relative forecasting accuracy of the examined data augmentation methods, we consider the following three benchmarks.

- **MLP-LW:** This method is implemented by employing the ensemble of feed-forward NNs described in Section 4.2. However, in MLP-LW, each time series contributes to the training set only its last available, in-sample window. Consequently, although the resulting train sets will contain the most recent in-sample observations from each series, they will be more limited in size. Note that since some of the series are too short to provide even

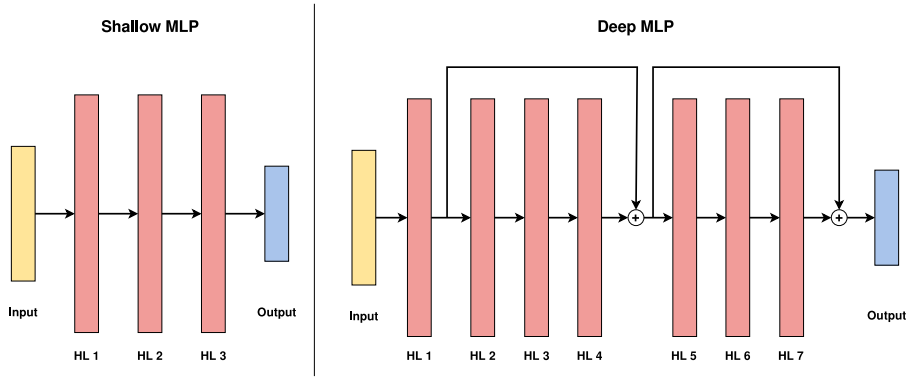

Fig. 2. The architecture of the shallow (left) and the deep (right) feed-forward networks used in the respective forecasting ensemble.

Table 2
The architecture and the training hyper-parameters of the feed-forward networks used in the forecasting ensemble, along with the considered search space for each hyper-parameter and the values selected as optimal by the optimization algorithm.

Hyper-parameter	Search space	Selected value
Input size (IS)	[6, 12, 18, 24, 30]	18
Number of hidden layers	[1, 2, 3]	3
Size of hidden layers	[0.5, 1.5, 2.0, 3.0, 5.0] ×IS	1.5 × IS
Optimizer	[Adam]	Adam
Learning rate	[0.0001, 0.0005, 0.001, 0.005, 0.01]	0.005
Number of training steps	[1000, 5000, 10,000, 20,000, 40,000]	20,000
Batch size	[64, 128, 256, 512, 1024]	512
Loss function	[MAE, MSE, sMAPE]	MAE

a single training sample, and no padding is being used, the resulting train sets contain 134, 1439, and 14,242 samples in the case of D_{Small}, D_{Medium}, and D_{Large}. Similarly, the corresponding training sets from the yearly, quarterly, and monthly data of the Tourism's competition consist of 419, 426, and 365 training samples, respectively.

- **MLP-AW:** Similar to MLP-LW, this method is also implemented by employing the ensemble of feed-forward NNs described in Section 4.2. In the case of MLP-AW, however, all available time series windows, extracted as described in Section 4.1, are being used for training the NNs. This slicing [19] approach is the easiest to implement for maximizing the number of training samples without using data augmentation. As a result, the corresponding train sets consist of 2544, 24,275, and 235,460 training samples in the case of D_{Small}, D_{Medium}, and D_{Large} sets. Similarly, the training sets from the yearly, quarterly, and monthly data of the Tourism's competition consist of 3231, 25,900, and 65,754 training samples, as shown in Table 1.
- **Theta:** Theta [1] is a well-established local statistical forecasting method, used in this study as a standard of comparison. After its introduction, Theta became popular for winning the M3 competition [22] and, later, for being the most accurate benchmark in the M4 competition [24].

4.4. Data augmentation

We consider the D_{Small}, D_{Medium}, and D_{Large} sets from the M4 yearly series and the Yearly, Quarterly, and Monthly sets from the Tourism competition along with their respective train sets, containing all available time series windows, as the "baseline" sets that will be augmented and then used for training purposes. In the literature, slicing the original series into multiple windows instead of extracting a single window per series is often viewed as a data augmentation technique in and of itself [13]. However, for the purposes of the present study we argue that data augmentation should refer to techniques that go beyond utilizing the maximum number of samples that can potentially be extracted from the original series. Thus, we distinguish slicing (MLP-AW) from the techniques described in Section 3 that alter the original data points or generate new.

For each set, we create synthetic samples so that the initial number of training samples doubles, i.e. we increase the number of the training samples of D_{Small} from 2544 to 5088, of D_{Medium} from 24,275 to 48,550, and of D_{Medium} from 235,460 to 470,920. Similarly, we increase the number of the training samples of Tourism's Yearly set from 3231 to 6462, of Tourism's Quarterly set from 25,900 to 51,800, and of Tourism's Monthly set from 65,754 to 131,508. Note that, in each experiment, only one of the data augmentation techniques described in Section 3 is employed. The resulting set, containing the synthetic samples along with the originals, is the final train set used by the NN models. Naturally, the new synthetic samples are also scaled using the mix-max transformation to a target range of $[0, 1]$, as described in Section 4.1.

When random noise injection is employed for augmenting the original data, the corresponding MLP trained on the augmented set will be noted as MLP-RNoise. Likewise, MLP-VFlip, MLP-HFlip, MLP-Comb, MLP-MWarping, MLP-Inter, MLP-Boot, MLP-Gen, and MLP-Upsampling will refer to the use of vertical flipping, horizontal flipping, time series combinations, magnitude warping, time series interpolation, bootstrapping, time series generation, and upsampling, respectively.

4.5. Forecasting accuracy measures

To measure the forecasting accuracy of the examined data augmentation techniques, as well as that of the benchmarks, we employ the mean absolute scaled error (MASE) [12], a widely used, scale-independent accuracy measure, defined as:

$$MASE = \frac{\frac{1}{h}\sum_{t=n+1}^{n+h}|y_t - f_t|}{\frac{1}{n-1}\sum_{t=2}^{n}|t_t - t_{t-1}|},$$

where f_t is the forecast of the examined approache at point t, y_t the actual value of the series at point t, h the forecasting horizon, and n the number of the available historical observations. The MASE, which was one of the official measures used in the M4 and the Tourism competitions for evaluation submissions is first computed for each series of the respective data set separately. Then, the calculated per-series performance is averaged across all 230, 2300, and 23,000 series for the D_{Small}, D_{Medium} and D_{Large} data sets to compute the overall accuracy of the examined forecasting approach in the respective set. Similarly, for the Tourism data set, MASE values are averaged across all 518, 427, and 366 series for the respective yearly, quarterly, and monthly data sets.

5. Results and discussion

The forecasting accuracy (MASE) of the MLP models that were fitted in the M4 data sets using the augmented train sets is summarized along with that of the selected benchmarks in Table 3. The second, third, and fourth columns of the table present the forecasting accuracy of the shallow MLP and correspond to the D_{Small}, D_{Medium}, and D_{Large} data sets from the M4 competition's yearly series, respectively. Accordingly, the fifth, sixth, and seventh columns of the table present the forecasting accuracy of the deep MLP for the cases of the D_{Small}, D_{Medium}, and D_{Large} data sets. The best result for each data set is highlighted with boldface.

The results reported in Table 3 indicate that increasing the size of the train set can significantly improve the forecasting accuracy of NN models. First, using all windows that can be extracted from the initial series (MLP-AW), instead of just the last available one per series (MLP-LW), improves accuracy by 10.4%, 6.9%, and 7.2% in the D_{Small}, D_{Medium}, and D_{Large} data sets, respectively when using the shallow MLP. The corresponding improvements of MLP-AW over MLP-LW, for the case of the deep MLP, are 15.9%, 11.8% and 11.6%. Second, further increasing the size of the train set by employing data augmentation techniques can result to even greater improvements. For instance, the maximum additional improvements reported for data augmentation over the MLP-AW benchmark for the shallow MLP in the D_{Small}, D_{Medium}, and D_{Large} data sets are 2.3%, 0.6%, and 0.7%, respectively. Similarly, for the deep MLP, the maximum improvements from applying data augmentation are 7.9%, 3.4%, and 1.7% for the D_{Small}, D_{Medium}, and D_{Large} sets.

For the case of the shallow MLP, all data augmentation approaches outperform both MLP-LW and the statistical benchmark, Theta, by a great margin of an average extent of about 7.5% and 12.0%, respectively. Nevertheless, Table 3 suggests that not all data augmentation techniques are equally effective in improving forecasting performance. For instance, MLP-VFlip, MLP-HFlip, MLP-RNoise, MLP-MWarping, MLP-Boot, and MLP-Gen fail to further reduce the forecast error of MLP-AW, with the only exceptions being the MLP-VFlip and MLP-MWarping techniques in the D_{Small} data set. Also, it is interesting that MLP-Inter significantly reduces the forecasting accuracy in the D_{Small} data set by 11.3% while it performs better in the D_{Medium}, and D_{Large} sets. On the positive side, MLP-Comb and MLP-Upsampling are consistently better than MLP-AW across all three data sets, achieving improvements of about 1%, on average. Finally, we find that the smaller the size of the initial data set is, the larger the realized improvements become. For example, although the accuracy improvements of MLP-Comb and MLP-Upsampling over the MLP-AW benchmark start at 2.3% and 1.3% in the D_{Small} data set, respectively, they reduce to 0% and 0.7% in the D_{Large} data set.

When a deep MLP is employed for forecasting, the accuracy improvements from utilizing data augmentation techniques on the initial training sets more significant. For the case of D_{Small}, all the examined techniques, except bootstrapping, provide 0.3% to 7.9% more accurate forecasts than MLP-AW. When D_{Medium} is considered, most augmentation techniques improve the forecasting accuracy by 0.6% to 3.4%. MLP-Boot, MLP-Inter, and MLP-MWarping are the only methods the do not improve accuracy, with MLP-Boot being as accurate as MLP-AW, MLP-Inter being 0.3% less accurate than MLP-AW, and MLP-MWarping being 0.9% less accurate. For the case of D_{Large} MLP-Gen, MLP-Inter, MLP-Comb, and MLP-Upsampling are 0.3%, 0.3%, 0.7%, and 1.7% more accurate than MLP-AW, while the remaining techniques do not improve the forecasting performance. Overall, the results for the case of the deep MLP are consistent with those of the shallow MLP. MLP-Comb and MLP-Upsampling are the top performing methods across all three sets of series, improving accuracy by 3.5% and 4.2%, respectively, compared to MLP-AW. As expected, the benefits of augmenting a data set are greater when the original set is smaller. Overall, applying data augmentation leads to greater accuracy improvements when the underlying forecasting NNs are deeper.

To analyze the significance of the reported improvements due to data augmentation over the three benchmarks, we employ the multiple comparisons with the best (MCB) test [18]. The test computes the mean ranks of the examined forecasting approaches across the three M4 data sets, as measured by MASE, and concludes whether their differences are statistically significant or not. Fig. 3 summarizes the results of the statistical test. If the intervals of two methods do not overlap, this indicates a statistically different performance. Thus, methods that do not overlap with the gray interval of the figures are considered significantly worse than the best performing method, and vice versa.

In the case of D_{Small}, we find that upsampling has the lowest mean rank, being however significant better only to MLP-LW and Theta when shallow NNs are used. All data augmentation techniques, as well as MLP-AW, are identified as similarly accurate. MLP-Comb ranks first for the deep MLP, with upsampling ranking second, and, the rest of the methods following identified as equally accurate, with the exception of MLP-LW. Note, however, that D_{Small} includes a very limited number of series, a factor that notably affects the width of the intervals and makes the distinction of statistically better methods challenging. A similar picture emerges from the evaluation on the 2300 series of D_{Medium}. MLP-Upsampling remains the most accurate method, having the lowest mean rank, for both NN architectures. In the case of the shallow MLP, upsampling provides significantly more accurate forecasts than MLP-Vflip, MLP-Gen, MLP-MWarping, MLP-HFlip, MLP-Boot, Theta, and MLP-LW. When the deep MLP is used, MCB results suggest that upsampling is significantly better than MLP-Hflip, MLP-Boot, MLP-VFlip, MLP-Inter, MLP-AW, MLP-MWarping, Theta, and MLP-LW. Finally,

Table 3
Overall forecasting accuracy, as measured by MASE, of the examined data augmentation techniques and the selected benchmarks across all 230 (D_{Small}), 2300 (D_{Medium}), and 23,000 (D_{Large}) yearly time series of the M4 competition, considering both a shallow and a deep MLP.

	Shallow MLP			Deep MLP		
	D_{Small}	D_{Medium}	D_{Large}	D_{Small}	D_{Medium}	D_{Large}
Benchmarks						
Theta	3.58	3.50	3.38	3.58	3.50	3.38
MLP-LW	3.46	3.31	3.18	4.21	3.65	3.37
MLP-AW	3.10	3.08	2.95	3.54	3.22	2.98
Data augmentation techniques						
MLP-VFlip	3.09	3.13	2.98	3.44	3.20	2.99
MLP-HFlip	3.11	3.13	2.98	3.33	3.20	2.99
MLP-RNoise	3.14	3.09	2.97	3.44	3.17	2.98
MLP-Comb	**3.03**	3.07	2.95	**3.26**	3.16	2.96
MLP-MWarping	3.08	3.17	3.00	3.28	3.25	3.03
MLP-Inter	3.45	3.07	2.95	3.53	3.23	2.97
MLP-Boot	3.11	3.11	2.96	3.54	3.22	2.98
MLP-Gen	3.14	3.09	2.97	3.46	3.18	2.97
MLP-Upsampling	3.06	**3.06**	**2.93**	3.27	**3.11**	**2.93**

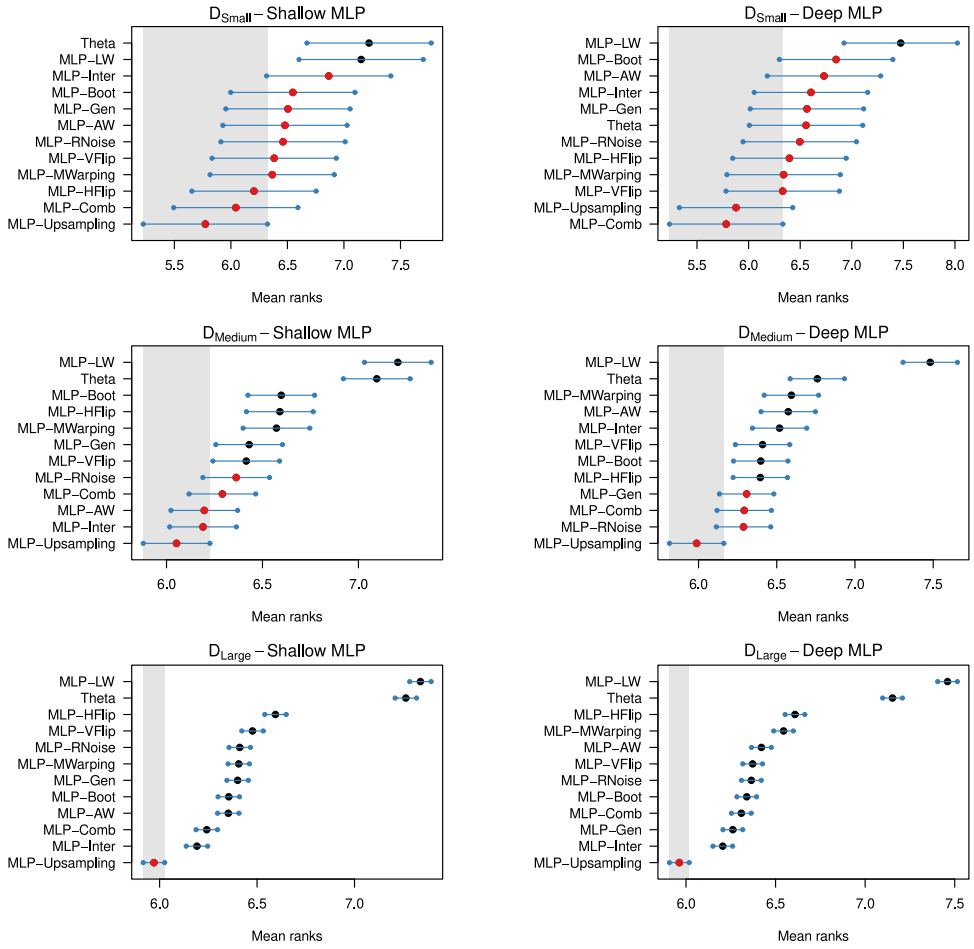

Fig. 3. Average ranks and 95% confidence intervals of the different data augmentation techniques and benchmarks considered over the 230, 2300 and 23,000 series of the D_{Small}, D_{Medium}, and D_{Large} data sets, respectively. The multiple comparisons with the best test, as proposed by [18], is applied using MASE for ranking the methods.

Table 4
Forecasting accuracy improvements from utilizing data augmentation techniques on the 518 yearly, 427 quarterly and 366 monthly series of the Tourism competition. Results for each augmentation technique are reported as percentage improvements over the benchmark approach of using all available original windows (MLP-AW), for shallow and deep MLPs.

	Shallow MLP			Deep MLP		
	Yearly	Quarterly	Monthly	Yearly	Quarterly	Monthly
MLP-VFlip	2.0%	0.0%	−1.3%	1.8%	1.1%	−2.5%
MLP-HFlip	0.0%	**1.9%**	0.0%	2.6%	**5.4%**	−0.6%
MLP-RNoise	**1.7%**	1.2%	**3.2%**	**3.2%**	3.3%	−0.6%
MLP-Comb	−1.3%	−3.7%	1.3%	−1.8%	2.2%	−0.6%
MLP-MWarping	−1.0%	−1.9%	0.6%	−2.4%	−3.8%	−1.3%
MLP-Inter	−1.7%	0.0%	1.3%	0.3%	2.2%	0.0%
MLP-Boot	−3.0%	−1.9%	−1.3%	0.9%	−1.1%	−5.1%
MLP-Gen	0.0%	−0.6%	0.0%	2.6%	0.5%	−0.6%
MLP-Upsampling	0.3%	1.2%	1.3%	2.9%	0.0%	**1.3%**

when all 23,000 series of D_{Large} are considered, MLP-Upsampling is considerably more accurate than the rest of the methods, for both NN topologies.

For the sake of brevity, Table 4 summarizes the percentage improvements from utilizing different data augmentation techniques over the MLP-AW benchmark for the yearly, quarterly, and monthly series of the Tourism competition. The results indicate that certain augmentation approaches can lead to accuracy improvements even in such a domain-specific forecasting application, irrespective of the frequency of the series, with MLP-RNoise, MLP-HFlip, and MLP-Upsampling being the most accurate approaches across all three data sets. When shallow NNs are used for producing the forecasts, MLP-RNoise, MLP-HFlip, and MLP-Upsampling provide, on average, 2.0%, 0.6% and 0.9% more accurate forecasts than MLP-AW. Similarly, when deeper NNs are employed, MLP-RNoise, MLP-HFlip, and MLP-Upsampling improve the forecasting accuracy of MLP-AW by 2.0%, 2.5% and 1.4% respectively. Finally, we finds that similar to M4, upsampling is among the top performing techniques in the Tourism competition's data sets, for both shallow and deep NNs.

Overall, our results suggest that increasing the number of the training samples can result in more accurate forecasts, especially when the size of the initial train set is limited. Such improvements can be realized either by extracting all possible windows from the original series or by employing data augmentation. Yet, our results

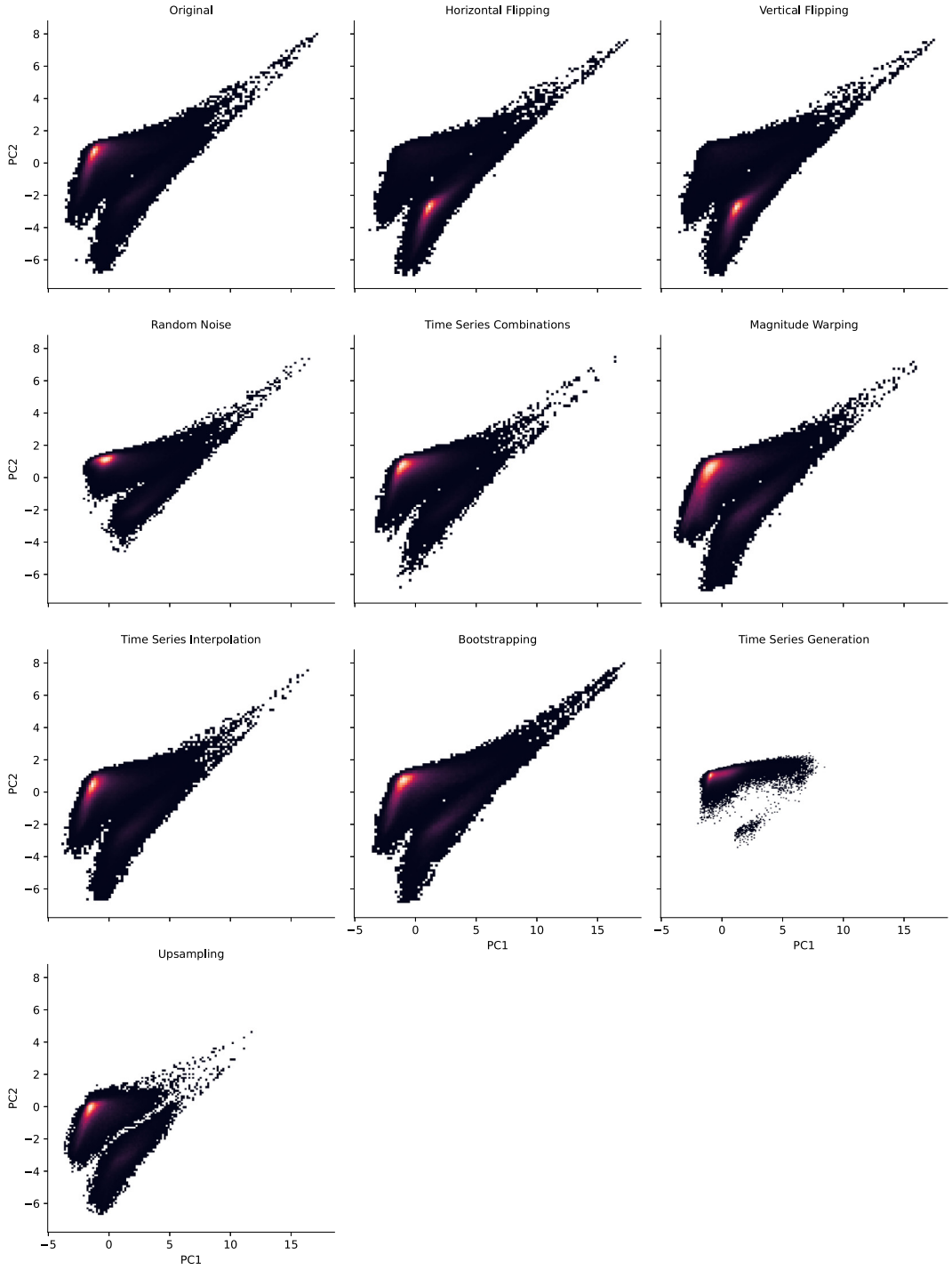

Fig. 4. Instance spaces of the training samples originally included in D_{Large} and those created by employing the examined data augmentation techniques. According to the coloring scheme used, brighter areas indicate higher concentration of samples with similar time series features.

indicate that accuracy improvements are subject to the data augmentation technique used. For instance, flipping, random noise injection, and magnitude warping were proven to be the least beneficial techniques on average, in contrast to upsampling and time series combinations that had a consistent positive impact on forecasting accuracy.

The variance in the forecasting performance of the different data augmentation techniques can be attributed to the characteristics of the synthetic samples each technique creates. In order for a data augmentation technique to become beneficial, it should be able to produce artificial series that are neither too similar to the original series nor too different than the series that the model will forecast when actually used, i.e. it should contribute new information to the forecasting model, while preserving the principles of the initial train set.

Augmentation approaches based on random adjustments, such as random noise injection, magnitude warping, and bootstrapping, create synthetic series that are very similar to the original ones, especially when the series exhibit strong trend, as it is the case with the yearly series of the M4 competition. Increasing the magnitude of the random component to enhance the diversity of the artificial data is not expected to mitigate this issue either since, by

doing so, the synthetic series will become too noisy and difficult to process. Similarly, augmentation techniques that create fundamentally different series are unlikely to positively affect forecasting accuracy, resulting in artificial data that do not necessarily represent the original series. For instance, in a set of series that are mostly governed by upward trends, as it is the case with the examined data sets, vertical and horizontal flipping would introduce mostly downwards trended series that are not representative of the original train set.

Combining or upsampling the series originally available in the train set seem to be more appropriate techniques since they avoid the issues mentioned above, while effectively increasing the number of the training samples. These techniques do not build on external random adjustments and ensure that the dominant characteristics of the original train set will be conserved. Time series combinations create synthetic series that consist of patterns that are present in the original series but scaled differently, resulting in adequately different samples. By focusing on a particular part of the original series, upsampling preserves and emphasizes local characteristics of the series and, as a result, the technique is able to generate new samples that are different but, at the same time, representative of the original samples.

To assess the aforementioned arguments, we investigate the characteristics of the original and the synthetic training samples created by the examined data augmentation techniques. First, we calculate nine indicative features that summarize key properties of the time series included in each data set, using the *tsfeatures* package for the R statistical software [11]. Then, we apply the visualisation method proposed by Kang et al. [16] that allows each time series to be represented in a two-dimensional instance space. In brief, the principal component analysis (PCA) is employed on the calculated time series features and, subsequently, the first two components, noted as PC1 and PC2, are used for placing the series in the instance space as a single point. Fig. 4 presents the instance spaces for the original and the synthetic samples of the D_{Large} data set. More details on the time series features considered, the visualisation method employed, as well as, the instance spaces of the D_{Medium} and D_{Small} data sets are provided as supplementary material.

We observe that both horizontal and vertical flipping maintain the outline of the distribution of the original series but significantly alter its density, as exhibited by the brighter areas of the respective distributions. The visual evidence suggest that the augmented set of series may not sufficiently represent the original data, leading to less accurate forecasts overall. A similar argument can be made for the time series generation technique. The corresponding plot indicates that although the generator provides series that cover some critical areas of the original instance space, the created series fail to cover a notable part of it. On the contrary, bootstrapping, magnitude warping, time series interpolation, random noise injection, and time series combinations produce samples that are very similar to the original data. Yet, density-wise, the time series combinations technique seems to produce series of more representative samples, a factor that may explain its superior forecasting performance over the rest of the examined techniques. Finally, by comparing the original data to those generated by upsampling, we find that the distribution of the series in the respective instance spaces are similar in areas that are more populous (leftmost parts of the scatter plots) but differ in areas that are more scarcely populated (rightmost parts of the scatter plots). Based on this observation, we conclude that balancing diversification and resemblance to the original series is a critical factor for improving forecasting performance when employing data augmentation.

6. Conclusions

This study investigated whether data augmentation can contribute to the development of more accurate NN models, specifically in the field of univariate time series forecasting. For that purpose, we implemented nine data augmentation techniques that cover different approaches in creating artificial data, ranging from simple transformations, adjustments, and interpolations to time series combinations and sophisticated generative models, including a new upsampling method that reported the most promising results overall. We assessed the corresponding forecasting performance of these techniques considering shallow and deep MLP networks and compared it to that of indicative benchmarks using several sets of series of different initial sizes, domains, and frequencies.

Our results suggest that increasing the number of the training samples can significantly improve the forecasting performance of NNs by up to 15%. However, we find that these improvements are dependent on the data augmentation technique used, being consistently positive for particular methods, such as upsampling and time series combinations, but negligible or even negative for others. Moreover, we find that data augmentation may be more beneficial to smaller data sets, given that the more the training samples are, the lower the realized accuracy improvements become. The same stands for deeper NNs compared to shallower ones since the generalization of the former type of networks typically requires significantly more data. Based on these findings and by examining the characteristics of the original versus the synthetic data constructed by each of the considered methods, we suggest that the development of advanced data augmentation techniques should focus on the creation of diverse, yet similar to the originals time series data and be mainly exploited in settings that involve relatively small train sets or deep NNs.

The findings of our study greatly motivate further research on data augmentation for univariate time series forecasting tasks. Some indicative, yet promising directions for expanding the scope of this study and tackling its current limitations are the following:

- In the present study, a large, diverse data set of 23,000 yearly series was used as the basis for conducting our experiments and evaluating the potential of data augmentation techniques along with a smaller data set originating from the tourism domain that consisted of monthly and quarterly series, among others. Nevertheless, it would be interesting to investigate whether our findings are confirmed for data sets that consist of higher frequency series (e.g. daily or hourly series) or cover other special forecasting applications (e.g. stock market, retail, and energy forecasting).
- Additional types of NN and ML models can be examined, apart from the MLP networks considered in this study. For example, RNNs treat input data as sequences of points introduced one after the other to the model, while CNNs process input data by sliding a small window across them. Thus, augmentation techniques that improve the accuracy of certain types of forecasting models may not necessarily add the same value to other types of models.
- More sophisticated approaches for generating synthetic series can be explored. Recent advances in deep learning algorithms have provided a large number of tools that can potentially be leveraged for time series forecasting. For example, generative adversarial networks (GANs), that have been used in time series analysis tasks, can be repurposed for creating artificial series that are representative of the original data.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.patcog.2022.109132

References

[1] V. Assimakopoulos, K. Nikolopoulos, The theta model: a decomposition approach to forecasting, Int. J. Forecast. 16 (2000) 521–530.
[2] G. Athanasopoulos, R.J. Hyndman, H. Song, D.C. Wu, The tourism forecasting competition, Int. J. Forecast. 27 (3) (2011) 822–844. Special Section 1: Forecasting with Artificial Neural Networks and Computational Intelligence Special Section 2: Tourism Forecasting
[3] K. Bandara, H. Hewamalage, Y.-H. Liu, Y. Kang, C. Bergmeir, Improving the accuracy of global forecasting models using time series data augmentation, Pattern Recognit. 120 (2021) 108148.
[4] C. Bergmeir, R. Hyndman, J. Benítez, Bagging exponential smoothing methods using STL decomposition and Box–Cox transformation, Int. J. Forecast. 32 (2) (2016) 303–312.
[5] J. Bergstra, B. Komer, C. Eliasmith, D. Yamins, D.D. Cox, Hyperopt: a python library for model selection and hyperparameter optimization, Comput. Sci. Discov. 8 (1) (2015) 014008.
[6] H. Cao, V.Y.F. Tan, J.Z.F. Pang, A parsimonious mixture of gaussian trees model for oversampling in imbalanced and multimodal time-series classification, IEEE Trans. Neural Netw. Learn. Syst. 25 (12) (2014) 2226–2239.
[7] S.F. Crone, M. Hibon, K. Nikolopoulos, Advances in forecasting with neural networks? Empirical evidence from the NN3 competition on time series prediction, Int. J. Forecast. 27 (3) (2011) 635–660.
[8] J.G. De Gooijer, R.J. Hyndman, 25 years of time series forecasting, Int. J. Forecast. 22 (3) (2006) 443–473.
[9] G. Forestier, F. Petitjean, H.A. Dau, G.I. Webb, E. Keogh, Generating synthetic time series to augment sparse datasets, in: 2017 IEEE International Conference on Data Mining (ICDM), 2017, pp. 865–870.
[10] H. Hewamalage, C. Bergmeir, K. Bandara, Global models for time series forecasting: asimulation study, Pattern Recognit. 124 (2022) 108441.
[11] R. Hyndman, Y. Kang, P. Montero-Manso, T. Talagala, E. Wang, Y. Yang, M. O'Hara-Wild, tsfeatures: Time Series Feature Extraction, 2022. R package version 1.0.2.9000.
[12] R.J. Hyndman, A.B. Koehler, Another look at measures of forecast accuracy, Int. J. Forecast. 22 (4) (2006) 679–688.
[13] Iwana, B.K. Uchida, Seiichi, An empirical survey of data augmentation for time series classification with neural networks, PLoS One 16 (7) (2021) 1–32.
[14] T. Januschowski, J. Gasthaus, Y. Wang, D. Salinas, V. Flunkert, M. Bohlke-Schneider, L. Callot, Criteria for classifying forecasting methods, Int. J. Forecast. 36 (1) (2020) 167–177.
[15] Y. Kang, R. Hyndman, F. Li, Gratis: generating time series with diverse and controllable characteristics, Stat. Anal. Data Min. 13 (2020).
[16] Y. Kang, R.J. Hyndman, K. Smith-Miles, Visualising forecasting algorithm performance using time series instance spaces, Int. J. Forecast. 33 (2) (2017) 345–358.
[17] L. Kegel, M. Hahmann, W. Lehner, Feature-based comparison and generation of time series, in: Proceedings of the 30th International Conference on Scientific and Statistical Database Management, in: SSDBM '18, Association for Computing Machinery, New York, NY, USA, 2018.
[18] A.J. Koning, P.H. Franses, M. Hibon, H.O. Stekler, The M3 competition: statistical tests of the results, Int. J. Forecast. 21 (3) (2005) 397–409.
[19] A. Le Guennec, S. Malinowski, R. Tavenard, Data augmentation for time series classification using convolutional neural networks, ECML/PKDD Workshop on Advanced Analytics and Learning on Temporal Data, 2016.
[20] P. Liu, X. Wang, C. Xiang, W. Meng, A survey of text data augmentation, in: 2020 International Conference on Computer Communication and Network Security (CCNS), 2020, pp. 191–195.
[21] K. Maharana, S. Mondal, B. Nemade, A review: data pre-processing and data augmentation techniques, Glob. Transit. Proc. 3 (1) (2022) 91–99. International Conference on Intelligent Engineering Approach(ICIEA-2022)
[22] S. Makridakis, M. Hibon, The M3-competition: results, conclusions and implications, Int. J. Forecast. 16 (4) (2000) 451–476.
[23] Makridakis, S. Spiliotis, E. Assimakopoulos, Vassilios, Statistical and machine learning forecasting methods: concerns and ways forward, PLoS One 13 (3) (2018) 1–26.
[24] S. Makridakis, E. Spiliotis, V. Assimakopoulos, The M4 competition: 100,000 time series and 61 forecasting methods, Int. J. Forecast. 36 (1) (2020) 54–74.
[25] C. Oh, S. Han, J. Jeong, Time-series data augmentation based on interpolation, Procedia Comput. Sci. 175 (2020) 64–71. The 17th International Conference on Mobile Systems and Pervasive Computing (MobiSPC),The 15th International Conference on Future Networks and Communications (FNC),The 10th International Conference on Sustainable Energy Information Technology
[26] B.N. Oreshkin, D. Carpov, N. Chapados, Y. Bengio, N-BEATS: neural basis expansion analysis for interpretable time series forecasting, CoRR abs/1905.10437(2019).
[27] F. Petropoulos, R.J. Hyndman, C. Bergmeir, Exploring the sources of uncertainty: why does bagging for time series forecasting work? Eur. J. Oper. Res. 268 (2) (2018) 545–554.
[28] F. Petropoulos, S. Makridakis, V. Assimakopoulos, K. Nikolopoulos, Horses for courses' in demand forecasting, Eur. J. Oper. Res. 237 (1) (2014) 152–163.
[29] D. Salinas, V. Flunkert, J. Gasthaus, T. Januschowski, DeepAR: probabilistic forecasting with autoregressive recurrent networks, Int. J. Forecast. 36 (3) (2020) 1181–1191.
[30] A.-A. Semenoglou, E. Spiliotis, S. Makridakis, V. Assimakopoulos, Investigating the accuracy of cross-learning time series forecasting methods, Int. J. Forecast. 37 (3) (2020) 1072–1084 July-September 2021.
[31] M. Shanker, M.Y. Hu, M.S. Hung, Effect of data standardization on neural network training, Omega 24 (4) (1996) 385–397.
[32] C. Shorten, T.M. Khoshgoftaar, A survey on image data augmentation for deep learning, J. Big Data 6 (1) (2019) 1–48.
[33] S. Smyl, A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting, Int. J. Forecast. 36 (1) (2020) 75–85.
[34] E. Spiliotis, A. Kouloumos, V. Assimakopoulos, S. Makridakis, Are forecasting competitions data representative of the reality? Int. J. Forecast. 36 (1) (2020) 37–53.
[35] A. Tealab, Time series forecasting using artificial neural networks methodologies: a systematic review, Future Comput. Inform. J. 3 (2) (2018) 334–340.
[36] T.T. Um, F.M.J. Pfister, D. Pichler, S. Endo, M. Lang, S. Hirche, U. Fietzek, D. Kulić, Data augmentation of wearable sensor data for Parkinson's disease monitoring using convolutional neural networks, in: Proceedings of the 19th ACM International Conference on Multimodal Interaction, in: ICMI '17, Association for Computing Machinery, New York, NY, USA, 2017, p. 216220.
[37] Q. Wen, L. Sun, F. Yang, X. Song, J. Gao, X. Wang, H. Xu, Time series data augmentation for deep learning: asurvey, in: Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, 2021.
[38] T. Wen, R. Keyes, Time series anomaly detection using convolutional neural networks and transfer learning, 2019. arXiv:2105.03075
[39] R. Ye, Q. Dai, Implementing transfer learning across different datasets for time series forecasting, Pattern Recognit. 109 (2021) 107617.
[40] J. Yoon, D. Jarrett, M. van der Schaar, Time-series generative adversarial networks, in: H. Wallach, H. Larochelle, A. Beygelzimer, F.d. Alché-Buc, E. Fox, R. Garnett (Eds.), Advances in Neural Information Processing Systems, vol. 32, Curran Associates, Inc., 2019.

Artemios-Anargyros Semenoglou is a Ph.D. student and research associate at the Forecasting & Strategy Unit of the National Technical University of Athens (NTUA). He graduated from the School of Electrical and Computer Engineering at the NTUA in 2017. His research interests are time series forecasting, data analytics and the use of neural networks and machine learning in operational research.

Evangelos Spiliotis is a Research Fellow at the Forecasting & Strategy Unit, National Technical University of Athens (NTUA), where he also serves as Coordinator. He graduated from School of Electrical and Computer Engineering, NTUA in 2013 and got his Ph.D. in 2017 from the same school. His research interests include time series forecasting, decision support systems, machine learning, optimisation, energy forecasting, energy efficiency and conservation. He has conducted research and development on tools for management support in many National and European projects. He co-organized the M4 and M5 Forecasting Competitions.

Vassilios Assimakopoulos is a professor at the School of Electrical and Computer Engineering of the National Technical University of Athens. He has worked extensively on applications of Decision Systems for business design and he has conducted research on innovative tools for management support. He specialises in various fields of Strategic Management, Design and Development of Information systems, Statistical and Forecasting Techniques. He coorganized the M4 and M5 Forecasting Competitions.