Layered Ensemble Architecture for Time Series Forecasting

Md. Mustafizur Rahman, Md. Monirul Islam, *Member, IEEE*, Kazuyuki Murase, and Xin Yao, *Fellow, IEEE*

***Abstract*—Time series forecasting (TSF) has been widely used in many application areas such as science, engineering, and finance. The phenomena generating time series are usually unknown and information available for forecasting is only limited to the past values of the series. It is, therefore, necessary to use an appropriate number of past values, termed lag, for forecasting. This paper proposes a layered ensemble architecture (LEA) for TSF problems. Our LEA consists of two layers, each of which uses an ensemble of multilayer perceptron (MLP) networks. While the first ensemble layer tries to find an appropriate lag, the second ensemble layer employs the obtained lag for forecasting. Unlike most previous work on TSF, the proposed architecture considers both accuracy and diversity of the individual networks in constructing an ensemble. LEA trains different networks in the ensemble by using different training sets with an aim of maintaining diversity among the networks. However, it uses the appropriate lag and combines the best trained networks to construct the ensemble. This indicates LEAs emphasis on accuracy of the networks. The proposed architecture has been tested extensively on time series data of neural network (NN)3 and NN5 competitions. It has also been tested on several standard benchmark time series data. In terms of forecasting accuracy, our experimental results have revealed clearly that LEA is better than other ensemble and nonensemble methods.**

***Index Terms*—Accuracy, base predictors, diversity, ensemble, time series forecasting (TSF).**

I. Introduction

TIME series forecasting (TSF) is the use of a model or technique to predict future values based on previously observed values [1]–[3]. It has been widely used in many real-world applications. For example, in finance, experts forecast stock market indices; in information technology, data processing specialists forecast the flow of information in their networks; in meteorology, experts predict the weather conditions of tomorrow. Generally, the phenomena generating a time series are unknown and information available for forecasting is only limited to the past values of the series. It is thus important to use an appropriate number of past values, termed lag, for forecasting.

Over the last few decades, there have been immense interests for understanding and predicting the future. This gives us many forecasting methods; most of them are relying on linear and nonliner statistical models. Linear statistical models are easy to explain and implement, but these models are inappropriate for time series originating from a nonlinear process. Although nonlinear statistical models [4]–[6] overcome some of the limitations of their linear counterparts, they are still limited in a way to solve real-world TSF problems [7]. Furthermore, the formulation of a nonlinear statistical model for TSF problems is a difficult task [8].

A multilayer perceptron (MLP) network, a kind of artificial neural network, has been widely used as a promising alternative approach to the forecasting society. It has been shown that the MLP network is a universal function approximator [9], but there exists no general guideline to choose the appropriate network architecture for solving a given problem. An ensemble brings together several individual networks (base predictors) for improving the generalization performance of a learning system [10]. It also alleviates the difficulty associated with the conventional design strategy of building a single best network with optimal parameters.

Although there have been many attempts in developing neural network ensembles for classification problems, there are only a handful attempts for TSF problems. The main issue in ensemble approaches is the consideration of accuracy and diversity of the individual networks used for constructing ensembles. Both theoretical [11] and empirical studies [12] have demonstrated that the generalization performance of the ensemble depends greatly on both accuracy and diversity among the networks. However, existing ensemble approaches for TSF problems consider either accuracy or diversity but not the both (Section II).

This paper proposes a layered ensemble architecture (LEA) for TSF problems. Our LEA consists of two layers, each of which uses an ensemble of MLP networks. The essence of the proposed architecture is that it considers both accuracy and diversity not only in generating the individual networks but also in combining the networks for producing the ensemble output. To encourage accuracy and diversity in the training and combination phases, LEA employs different techniques.

The rest of this paper is organized as follows. Section II discusses state-of-the-art ensemble algorithms for TSF problems. Section III describes our LEA in detail. In Section IV, we

Manuscript received November 11, 2014; revised January 25, 2015; accepted January 31, 2015. Date of publication February 24, 2015; date of current version December 14, 2015. This paper was recommended by Associate Editor L. Zhang.

Md. M. Rahman and Md. M. Islam are with the Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Dhaka 1000, Bangladesh (e-mail: mdmonirulislam@cse.buet.ac.bd).

K. Murase is with the Department of Human and Artificial Intelligence Systems, University of Fukui, Fukui 910 - 8507, Japan.

X. Yao is with the School of Computer Science, University of Birmingham, Birmingham B15 2TT, U.K.

This paper has supplementary downloadable multimedia material available at http://ieeexplore.ieee.org/ provided by the authors. This includes a PDF file that contains additional tables not included within the paper itself. This material is 66 KB in size.

Digital Object Identifier 10.1109/TCYB.2015.2401038

present the experimental procedures and detail evaluations and comparisons of LEA. We briefly describes the sensitivities of different parameters of LEA in Section V. Finally, Section VI concludes this paper with a few remarks for future directions.

II. Related Work

Combining several base predictors for improving forecasting accuracy has been extensively studied in the traditional forecasting literature. Clemen [13] provides a comprehensive review and annotated bibliography in this area. The aim of this section is to describe computational intelligence (CI)-based ensemble approaches for TSF problems.

As mentioned before, consideration of diversity and accuracy of the networks is the cornerstone of ensembles. Diversity can be ensured using different training sets, different parameters, or different types of networks. Accuracy of the predictors can be ensured using the appropriate lag and optimal parameters. Specifically, the lag parameter is the most critical component in forecasting. This is because it corresponds to the number of past observations used by the networks to capture the underlying characteristics of a given time series. If the lag is small, important information may be left out. On the other hand, useless inputs may introduce noise, if the lag is large. Both of these situations are not beneficial for obtaining a good forecasting accuracy [14], [15].

Boosting [16] and bagging [17] are two popular ensemble methods that use a resampling technique to create different data sets for training different networks in an ensemble. In [18], an ensemble constructed by feed-forward neural networks and trained by boosting is proposed for TSF problems. Boosting is also coupled with the recurrent neural networks in [19] and [20]. Instead of using neural networks as base predictors (see [21], [22]) employ genetic programming as the base predictors.

The application of bagging is also found like boosting in the forecasting literature. Zheng [23] used bagging with neural networks for binary prediction of financial time series. Inoue and Kilian [66] explore the usefulness of bagging in forecasting economic time series from the perspective of the linear multiple regression models. Bagging with competitive associative networks is applied to the time series data of the neural network (NN)3 forecasting competition [24]. In [14], bagging and random subspace [25] are used for constructing an ensemble. The authors use here decision trees as the base predictors.

Different data sets can be created without using boosting, bagging or random subspace. For example, Ruta *et al.* [31] created different data sets by randomly permuting, partitioning, and injecting noise to the original time series data. They first train k groups of networks. Each group contains on average M/k networks. The best k networks, one from each group, are used for constructing the ensemble. Zhang and Berardi [15] used the lag parameter for partitioning the time series data to create different training sets.

Instead of using different training sets, there are several works that vary the parameters of the networks in constructing ensembles. For example, Barrow *et al.* [32] investigated the performance of two different types of ensembles. One type varies the initial weights of the networks, while the other type varies the architecture of the networks. In [26], an ensemble of "echo state" networks, a special case of recurrent neural networks [27], with different memory lengths is proposed. This method secures the first position in the NN3 forecasting competition. An ensemble consisting of three radial basis function (RBF) networks is proposed in [28]. This method is applied on the time series data of the NN3 forecasting competition, where the spread factors of the RBF networks are set to the 50th, 75th, and 95th percentiles of the nearest distances of all training samples to the rest of the points.

All the ensemble approaches discussed so far combine the same type of networks for constructing ensembles. It is, however, possible to construct ensembles by combining different types of networks. For example, Wichard and Maciej [67] combine base predictors taken from linear and polynomial models, k-nearest neighbors, MLP networks, and RBF networks. Unlike [29], Wolpert [30] used RBF networks, k-nearest neighbors, and self organizing maps for constructing ensembles.

A careful scrutiny of existing ensemble approaches reveals that all these approaches except [28] emphasize only on the diversity of the base predictors. Diversity in the existing approaches is encouraged using different data sets for the base predictors (see [14], [18], [19], [31]), different parameters for the base predictors (see [26], [32]), and different types of base predictors (see, [29], [30]). Not only that, aside from [31], all other works do not consider accuracy and diversity in combining predictors for obtaining the ensemble output. Although the method proposed in [28] ensures accuracy of the RBF networks using an appropriate lag, the networks will be less diverse as they differ only by the spread. It has been known that training networks using different data is more effective for maintaining diversity [33]–[35]. Furthermore, the method proposed in [28] finds the appropriate lag by employing a trial-and-error method.

III. LEA

In order to reduce the detrimental effect of using a predefined lag and to devise a good forecasting scheme, a layered architecture is proposed for TSF. Our proposed architecture, LEA, consists of two ensemble layers. While the first ensemble layer tries to find an appropriate lag of a given time series, the second ensemble layer employs the obtained lag for forecasting. It has been known that the performance of any forecasting model is greatly dependent on the lag [14], [15], [20], [28], [31]. That is why a layered architecture is adopted in this paper.

The major steps of LEA can be described in Fig. 1, which are explained further as follows.

1) Preprocess data points (observations) of a given time series for handling noise, missing attribute(s), and seasonality.
2) Hold out m data points for testing LEA and use the remaining data points for constructing LEA.
3) The first ensemble layer is created as follows.

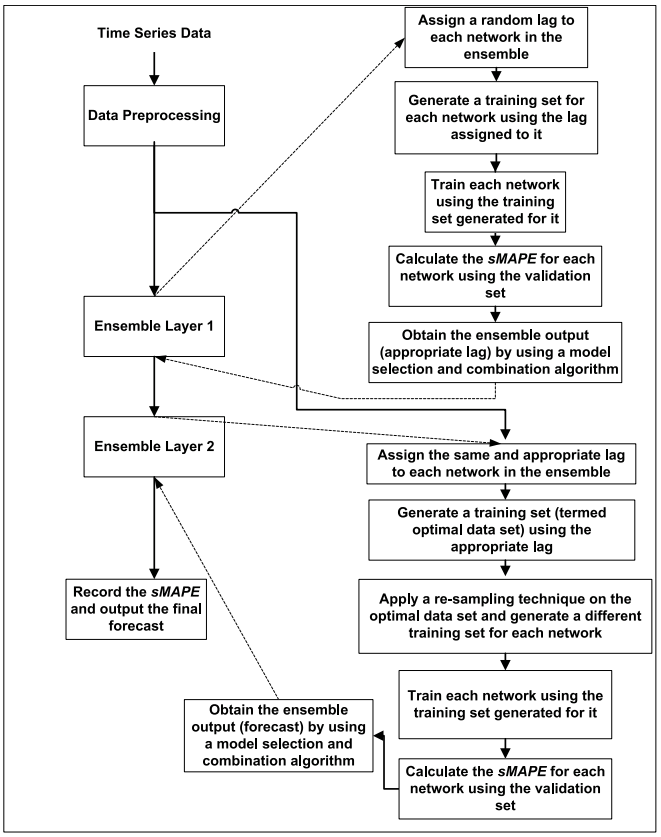

Fig. 1. Major steps of LEA.

Algorithm 1 Model Selection and Combination Algorithm

Require: Ensemble size τ, cluster size k.

for $i = 1$ to τ **do**
  Calculate the sensitivity of each MLP.
end for
Cluster the τ MLPs into k clusters ($k < \tau$) based on their sensitivities.
Select one MLP from each cluster which has the lowest sMAPE in that cluster over the validation set.
if Ensemble Layer 1 **then**
  Record the assigned lag of each selected MLP.
  Provide the final lag by averaging the lags of the selected MLPs and applying flooring on the average value to obtain an integer value.
else
  if Ensemble Layer 2 **then**
    Calculate the combination weight for weighted average using the sMAPE of each selected MLP.
    For each selected MLP perform out-of-sample prediction.
    Provide the final out-of-sample forecast using weighted average.
  end if
end if

a) Create an ensemble consisting of τ MLP networks. Here τ is a user-defined parameter and greater than $l_{\max}$, the maximum lag of the time series. For example, $l_{\max}$ can be 12 for a monthly time series.
b) Assign a random lag, l_i, to each network i in the ensemble. This assignment is done by uniformly generating a random number between 1 and $l_{\max}$.
c) Define the architecture of each network in the ensemble. The architecture consists of an input layer, a hidden layer and an output layer. The number of nodes in the input and hidden layers equals to the lag assigned to the network, while the number of nodes in the output layer is one.
d) Generate τ training sets, one for each network, using the lags assigned to all τ networks in the ensemble.
e) Train each network i in the ensemble on the training set generated for it using the Levenberg–Marquardt (LM) algorithm [36].
f) Evaluate all the trained networks on a validation set containing P data points. The symmetric mean absolute percent error (sMAPE) can be used for evaluation. According to [37], sMAPE can be expressed as

$$\text{sMAPE} = \frac{1}{P}\sum_{p=1}^{P}\frac{|Y(p) - \hat{Y}(p)|}{(Y(p) + \hat{Y}(p))/2} \times 100 \qquad (1)$$

where $Y(p)$ and $\hat{Y}(p)$ are the actual output and the predicted output on the pth validation example, respectively. The smaller the sMAPE is, the better the forecasting accuracy is. We use sMAPE because it is used (see [14], [28]) and forecasting competitions (e.g., NN3 [37], NN5 [38]). However, any other performance measure can be used for evaluating the network.
g) Obtain the ensemble output by applying the model selection and combination method (Algorithm 1) on the trained networks. As the objective of the first ensemble layer is to find the appropriate lag, the ensemble output of this layer is lag.

4) The second ensemble layer is created as follows.
a) Create an ensemble consisting of τ MLP networks.
b) Assign the same and appropriate lag, which is obtained by the first ensemble layer, to all the τ networks.
c) Define the architecture of the networks in the same way as described in step 3c). Note that the architecture of all networks in this ensemble layer is same. This is because the lag assigned to the networks is same and it is used in defining the architecture of the networks.
d) Generate an optimal data set, D_{tr}, using the appropriate lag.
e) Train each network i in the ensemble on a training set, D_{tr}^{i}, using the LM algorithm. Our LEA generates D_{tr}^{i} by resampling $k\%$ data of D_{tr}. For example, bagging [17] or boosting [16]) can be used for resampling.

f) Obtain the ensemble output by applying the model selection and combination method (Algorithm 1) on the trained networks. Unlike the first ensemble layer, the output of this ensemble layer is forecast.

The above LEA appears to be straight forward. However, its essence is the techniques incorporated for maintaining accuracy and diversity among the networks in the ensembles. Our LEA also has some other components. Details of them are discussed in the following sections.

A. Data Preprocessing

Our data preprocessing includes noise removal, missing attribute(s') treatment, and handling seasonality (deseasonalization).

1) Noise Removal: In many TSF problems (e.g., NN3 [37] and NN5 [38]), the data points in the time series are heavily influenced by noise. A data point is considered as noise if its value is very much different from other values of the series. Failure to take specific measures against noise may lead to a bad forecasting performance.

There are several ways by which one can remove noise from the time series. For example, a large second order difference is used as an indication of noise [39]. In LEA, we use a noise detection mechanism proposed in [28]. A data point, d_i in this mechanism is considered as noise if its absolute value is four times greater than the absolute medians of the three consecutive points before and after that point. That is, d_i is noise if its value satisfies the condition: $d_i \geq 4 \times \max\{|m_a|, |m_b|\}$, where $m_a = \text{median}(d_{i-3}, d_{i-2}, d_{i-1})$ and $m_b = \text{median}(d_{i+3}, d_{i+2}, d_{i+1})$. When the data point is identified as noise, its value is simply replaced by the average value of the two points that are immediately before and after it.

2) Missing Value Treatment: A time series data may contain some missing value. Further preprocessing is necessary in such a situation. For example, the NN5 [38] time series data requires a preprocessing step, termed gap removal. Two types of anomalies exist in the NN5 time series data. One is zero value that indicates no money withdrawal occurred at a particular time. Another is missing observations for which no value is recorded. A gap removal strategy tries to identify those anomalies and removes them. In this paper, we adopt the gap removal strategy proposed in [40]. If d_i is a gap, this strategy replaces the gap with the median of a set [d_{i-365}, d_{i+365}, d_{i-7} and d_{i+7}]. Here all or the available members of the set are used for computing the median.

3) Handling Seasonality: Handling seasonality i.e., deseasonalization is one of the major issues in the TSF literature. Many time series (e.g., NN3 [37] and NN5 [38]) contain some sorts of seasonality. Two main methods for handling seasonality are direct and deseasonalized [41]. In the direct method, the base predictors are trained directly on the raw data, whereas in the deseasonalized method, seasonal adjustments are made on the raw data before the networks are trained on.

The proposed LEA applies a simple deseasonalization procedure suggested in [42] on the seasonal time series of NN3. This procedure simply subtracts the seasonal average from the series to obtain a deseasonalized series. The time series of NN5 possess a variety of periodic patterns. LEA applies the deseasonalization method described in [41] on the time series of NN5 to remove the strong day of the week seasonality as well as the moderate day of the month seasonality. To make the final forecast, LEA restores back the seasonality.

B. Training Set Generation

A training set in the form $\{X(t); Y(t)\}_{t = 1 \text{ to } T}$ is necessary for obtaining the optimal or near optimal weights of an MLP network. Here $X(t)$ represents an input to the network and may have several components i.e., $X(t) = d_q, \ldots, d_{q+l}$. We assume that the output $Y(t)$ has one component.

To generate a training set for a given time series, we need some extra effort because data points of the series are only available. The parameter needed in generating such a set is the lag, l. Let l is 5 and the data points $d_1, d_2, \ldots, d_r$ of the series are used for generating the training set. The generation process takes l as a window and shifts it in generating the training set. That is, $X(1) = d_1, d_2, \ldots, d_5$ and $Y(1) = d_6$, $X(2) = d_2, d_3, \ldots, d_6$ and $Y(2) = d_7$ and this process continues until $Y(t)$ reaches at the end of the series i.e., d_r. It is now clear that we can get a different training set by using a different lag. This is why LEA assigns different lags to the networks of the first ensemble layer [step 3b)].

LEA employs two steps to generate the training sets for the networks in the second ensemble layer. In the first step, it generates an optimal dataset D_{tr} by using an appropriate l, which LEA obtains by using its first ensemble layer. In the second step, LEA applies a bootstrapped resampling technique (e.g., bagging [17] or boosting [16]) τ times on D_{tr} and generates τ training sets, one for each network of the second ensemble layer.

C. Base Predictors

The MLP networks are flexible computing frameworks that are used as base predictors in many ensemble approaches. In its current implementation, LEA uses MLP networks as base predictors for the first and second ensemble layers. This is, however, not an inherent constraint. In fact, any type of networks (e.g., RBF network, recurrent neural network) can be used in LEA.

It is necessary to specify the architecture of the MLP network for using it. Specifically, the number of hidden layers and nodes therein determines the ability of the network. It has been shown that the MLP network with one hidden layer is universal approximator [9]. Since an ensemble is usually constructed by combining weak learners [16], [43], the exact determination of the network architecture is not essential. It has been shown that the in-sample and out-of-sample forecasting ability of the network are not very sensitive to the number of hidden nodes [44]. For the sake of simplicity, LEA thus keeps the number of hidden nodes same as the number of input nodes and determines this number by the lag of a given time series. Our LEA assigns different lags to the networks of the first ensemble layer, but it assigns the same lag to the networks of the second ensemble layer. This assignment causes a different ensemble for the two layers of LEA.

D. Model Selection for Ensemble

An important goal of a good ensemble is to improve the generalization performance of a learning system by combining a set complementary weak networks so that one network can offset the deficiency of the other network(s). The construction of the ensemble is usually divided into two phases [45]. A set of networks is first produced. Several accurate and diverse networks are then combined to obtain the ensemble output [46], [47]. Variance is usually used in regression problems to select the diverse networks [48] and simple averaging or weighted averaging is widely used for combination [49].

We here improvise a clustering-based model selection and combination method to obtain a better forecast by LEA. The essence of our method is that it considers both diversity and accuracy in combining the trained networks. The diversity among the networks in an ensemble is essentially attributed to the difference of their outputs. We use the networks' sensitivity as a measure to evaluate their diversity. The sensitivity may reflect the output behaviors of the networks not only for any particular instance, $X(p)$, but also for near that instance, $X(p) + \Delta X(p)$. This is why we perturb the instances in the validation set by adding noise.

Our LEA first calculates the outputs of a trained network on the instances of the validation set. It then adds noise on the instances of the validation set and calculates the sMAPE on this noisy set. Finally, LEA takes the average difference of the outputs and define it as the sensitivity, S, of the network. That is

$$S = \frac{1}{P}\sum_{p=1}^{P} |Y(X(p)) - Y(X(p) + \Delta X(p))|. \tag{2}$$

For adding noise to an instance, LEA simply adds a value to the feature value of the instance. This value is randomly generated by the Gaussian distribution with mean zero and standard deviation one. It has been known that Gaussian distribution has a higher probability than Cauchy distribution for producing a small value [50], which is suitable for producing the nearby instance. However, it is possible to use uniform distribution with a small $|b - a|$ so that the two instances $X(p)$ and $X(p) + \Delta X(p)$ are similar. Noise is almost always normally distributed in all measurements under very general conditions, aside from unusual cases that involve a nonlinear bias in the measurement, or nonlinearity in the system itself. The nonlinear phenomena are difficult to study for many reasons. Hence, the vast majority applications deal with the linear phenomena, which is also followed in this paper.

There are τ sensitivities as LEA trains τ networks for the first or second ensemble layer. The most well known k-means clustering algorithm is applied on these sensitivities to select accurate and diverse trained networks. It is important to note that LEA finds the value of k by adopting an incremental approach. Our algorithm selects the k best networks, one from each cluster and combines these networks to construct an ensemble. The use of the sensitivities and the selection of the best networks indicate LEAs emphasis on diversity and accuracy in the combination phase.

Since the objective of the first ensemble layer is to find the appropriate lag of a time series, the lags of the best networks selected for this layer are recorded. LEA first takes the average of these lags and then applies flooring on the average value to obtain the appropriate lag. Flooring is applied here for making the average value to an integer one. For the second ensemble layer, LEA combines the forecasts of the best networks by employing the weighted average. The weight of a network is inversely proportional to its corresponding sMAPE over the validation set. As forecast can have a real value, we do not require here flooring. The pseudo code of the whole process is given in Algorithm 1.

E. Accuracy and Diversity

There is a considerable amount of evidence that an ensemble of MLP networks can lead to a improved generalization performance [51]. Both theoretical [11] and empirical studies [12] have demonstrated that the generalization performance of an ensemble is greatly dependent on both diversity and accuracy of individual MLP networks. Our LEA ensures diversity by employing a different technique in its first and second ensemble layers. For example, it assigns different lags to different networks of the first ensemble layer. We have already discussed that the use of different lags leads to different training sets (Section III-B). Although LEA assigns the same lag to different networks of the second ensemble layer, it uses a resampling technique for generating different training sets. It has been known that training networks using different training sets is the most effective way for producing diverse trained networks [34], [52].

The use an appropriate lag of a time series is important for ensuring accuracy of the networks. Unfortunately, such a lag is not known in advance and may be different for different time series. At the beginning of forecasting, we only have time series. What can we do here? We can first train a network several times using the training sets generated by employing all the possible lags of the series and then select a lag for which the network produces the best forecast. Instead of using such an approach, LEA uses an ensemble of networks i.e., the first ensemble layer for determining the appropriate lag. It trains τ networks for constructing the ensemble where $\tau >> l_{\max}$. That is, more than one networks will have the same architecture and the training set. This gives a sort of redundancy for the first ensemble layer, an indirect way for emphasizing on accuracy of this layer's networks. However, the second ensemble layer directly emphasizes on accuracy by using the appropriate lag obtained by the first ensemble layer. Moreover, the first or second ensemble layer construct the ensemble by combining the outputs of the k best and diverse networks selected from a set of τ trained networks where $k < \tau$. All these techniques are incorporated to ensure the diversity and accuracy of the networks in the ensemble.

An important question now arises how our choice of "accuracy and diversity" necessarily guarantees the generalization

performance in the context of TSF. According to [48], the ensemble generalization error, E, can be expressed as

$$E = \bar{E} - \bar{A}. \tag{3}$$

The first term, $\bar{E}$, is the weighted average of the generalization errors of the individual networks that constitute an ensemble and the second term, $\bar{A}$, is the weighted average of the ambiguities (i.e., diversities) between the networks. To get a smaller E, it is necessary to have a smaller $\bar{E}$ and a larger $\bar{A}$.

We use MLP networks with one hidden layer as the individual networks of an ensemble. The hidden layer of such a network contains h number of nodes. The one-step ahead forecast $\hat{Y}$ is computed using the inputs of a lagged observation $X = d_1, d_2, \ldots, d_l$. Here, l denotes the lag parameter of a time series. We can express the forecast as

$$\hat{Y} = \beta_0 + \left(\sum_{j=1}^{h} \beta_j \times f_h \left(\alpha_{j0} + \sum_{i=1}^{l} \alpha_{ji} d_i \right) \right). \tag{4}$$

In (4), $W = (\beta, \alpha)$ are the network weights with $\beta = \beta_1, \beta_2, \ldots, \beta_h$ representing the weights connecting to the output node and $\alpha = \alpha_{11}, \alpha_{12}, \ldots \alpha_{hl}$ representing the weights connecting to the hidden nodes. The β_0 and α_{j0} are the biases of the output node and each hidden node, respectively. The variable f_h is a nonlinear transfer function employed for the hidden nodes.

A supervised learning algorithm (e.g., back-propagation or LM) updates the weights during training so that $\hat{Y}$ becomes very close to the actual output, Y. The amount of the weights' change by the algorithm is given by

$$\Delta W \propto X. \tag{5}$$

As ΔW is dependent on X, the weights learned for different networks will be different if we train the networks using different data sets (input vectors). The outputs of the networks in such a situation will be different for the same input. That is, the networks will be diverse and $\bar{A}$ will be large. On the other hand, if the input vectors does not contain proper information due to inappropriate l, the weights learned for the networks will be inappropriate. The result is a large $\bar{E}$. In our LEA, we use different data sets for achieving a large diversity and an appropriate l for achieving a good accuracy i.e., small $\bar{E}$. Thus our choice of accuracy and diversity is clearly able to produce an improved ensemble generalization.

F. Computational Complexity of LEA

This section presents a brief insight about the computational complexity of LEA. In addition to traditional ensemble learning, LEA adds noise to the data and applies clustering in producing the ensemble output.

1) *Ensemble Learning:* LEA trains τ MLP networks for constructing ensembles like traditional bagging and boosting.
2) *Addition of Noise:* Since LEA simply adds Gaussian noise, the runtime of noise generation, and addition is linear with the size of the validation set, for example, $O(P)$. Here P is the number of examples in the validation set.
3) *Clustering:* The runtime of clustering is $O(k\tau)$, where k is the number of clusters and τ is the number of sensitivities obtained from the trained networks.

The runtime of LEA is, thus, the summation of three components: runtime of traditional ensemble learning, $O(P)$ and $O(k\tau)$. The summation of $O(P)$ and $O(k\tau)$ would be much less than the runtime of ensemble learning. It is now clear that the approximate runtime of our layered ensemble approach is almost same as the traditional ensemble learning method.

TABLE I
NUMBER OF TIME SERIES IN EACH CATEGORY OF THE NN3 FORECASTING COMPETITION [37]

	short	long	Sum
non-seasonal	25	32	57
seasonal	25	29	54
Sum	50	61	111

IV. EXPERIMENTAL STUDIES

This section first presents the detail performance evaluation and comparison of LEA using the time series data of the NN3 forecasting competition [37]. We then briefly present the comparisons of LEA using the time series data of the NN5 forecasting competition [38] and several traditional time series data.

The NN3 competition provides two datasets A and B. The dataset A contains 111 monthly time series drawn from a homogeneous population of empirical business time series, while the dataset B contains only 11 series taken from A. These datasets and their detail description can be found from http://www.neural-forecasting-competition.com/NN3/datasets.htm. We use here the dataset A. Organizers of the NN3 competition categorize the time series into long and short based on the series length. According to the characteristics of data, the time series can be further categorized into seasonal and nonseasonal. Table I shows the number of series in each category.

A. Performance Measure

The global performance of a forecasting model is usually evaluated by some accuracy measure such as sMAPE [53], median root absolute error (MdRAE) [54], and mean absolute scaled error (MASE) [54]. sMAPE is used in the NN3 [37], NN5 [38], and NN GC1 [55] forecasting competitions. One advantage of sMAPE is its scale independence property, which is suitable for comparing different methods across various series. Although sMAPE has been originally proposed in [56], most of the authors adopt the variant proposed in [53] that does not lead to negative values. We also use this variant in this paper.

A particular performance measure may favor the forecasting model if the measure implicitly or explicitly satisfies the

assumption or condition of the model. Hence, in order to make exhaustive evaluation, we use MdRAE and MASE in addition to sMAPE. The MdRAE and MASE performance measures can be expressed as

$$\text{MdRAE} = \text{median}(|r_n|), \quad r_n = \frac{Y(n) - \hat{Y}(n)}{Y(n) + \hat{Y}(n)^*} \tag{6}$$

$$\text{MASE} = \frac{\sum_{n=1}^{N} |Y(n) - \hat{Y}(n)|}{\frac{N}{N-1} \sum_{n=2}^{N} |Y(n) - Y(n-1)|} \tag{7}$$

where $\hat{Y}(n)^*$ is the forecast made by a reference method i.e., random walk [56] applied on the series data of a given forecast horizon, h. The main problem of MdRAE is that it returns infinite for equal consecutive observations [54]. Armstrong and Collopy [56] suggested that MdRAE is appropriate only for a very small set of series and it has a better protection against noise. The performance measure MASE scales the error based on the in-sample mean absolute error from the naïve method and is scale independent. Hyndman and Koehler [54] and Mager *et al.* [57] recommend MASE to become the standard measure for forecasting accuracy.

B. Experimental Setup

We use the tan-sigmoid function as the activation function for the nodes in the hidden layer of an MLP network. We set the learning rate and momentum term of the LM algorithm to 0.1 and 0.4, respectively. We stop a training process after 1000 epochs or when the root mean square error of the MLP network reaches to 0.00001. The ensemble size, τ, is set to 50. It is important to note that, we use the aforementioned parameter settings for all time series and these are not meant to be optimal.

According to [37], we keep the last 18 data points of every time series of NN3 for testing and use the remaining data points for building forecasting models. More specifically, from the remaining data points, we use 80% data points for training and 20% data points for validation. We normalize each data point of the series to zero mean and unitary variance. The normalization parameters are first computed from the training and validation data points. These parameters are then applied to the training, validation and testing data points.

We implement LEA for TSF problems by using MATLAB (R2012a, The Mathworks, Inc., Natick, MA, USA). The source codes of LEA are available from Md. M. Rahman (mustafiz_rahman@cse.buet.ac.bd).

C. Results

We first compare our LEA with basic bagging and boosting to show the effect of layering on the performance of these popular ensemble models. We then compare LEA with several other ensemble, nonensemble, and benchmark statistical methods.

1) Comparison With Bagging: As mentioned in Section III, LEA resamples k% data points to create different training sets for different networks in the second ensemble layer. Bagging is used here as a method for resampling. We name this version of LEA as layered bagging. The value of k is set to 9% for LEA.

TABLE II
APPROPRIATE LAG OBTAINED BY THE FIRST ENSEMBLE LAYER OF LEA FOR 111 TIME SERIES DATA OF THE NN3 FORECASTING COMPETITION [37]

		seasonal	non seasonal	short	long
LEA	mean	7.3529	6.7674	7.0600	7.1803
	minimum	1	1	1	1
	maximum	12	12	12	12
	std	3.2816	3.3441	3.3649	3.2788

TABLE III
COMPARISON BETWEEN BASIC BAGGING AND LAYERED BAGGING BASED ON AVERAGE SMAPE, MASE, AND MDRAE FOR SEASONAL, NONSEASONAL, SHORT AND LONG TIME SERIES DATA OF THE NN3 FORECASTING COMPETITION [37]. HERE, THE BEST RESULT IS HIGHLIGHTED USING BOLDFACE TEXT

		Basic Bagging			Layered Bagging		
		sMAPE	MASE	MdRAE	sMAPE	MASE	MdRAE
seasonal	mean	14.220	1.380	0.618	**13.270**	**1.242**	**0.546**
	min	0.875	0.484	0.206	**0.806**	**0.442**	**0.185**
	max	**60.584**	8.324	1.291	61.140	**4.886**	**1.096**
	std	**11.784**	1.174	0.246	12.023	**0.907**	**0.236**
non seasonal	mean	17.963	0.940	0.694	**16.612**	**0.686**	**0.571**
	min	5.819	0.499	0.289	**4.198**	**0.296**	**0.204**
	max	**76.622**	2.516	2.227	81.617	**1.724**	**0.977**
	std	13.703	0.501	0.340	**13.633**	**0.273**	**0.209**
short	mean	14.333	0.960	0.597	**13.599**	**0.755**	**0.500**
	min	5.819	0.499	0.206	**4.198**	**0.296**	**0.185**
	max	**40.682**	2.523	2.227	44.714	**2.378**	**0.977**
	std	**8.226**	0.535	0.348	8.423	**0.356**	**0.213**
long	mean	16.766	1.415	0.689	**15.357**	**1.249**	**0.601**
	min	0.875	0.484	0.273	**0.806**	**0.442**	**0.204**
	max	**76.622**	8.324	1.291	81.617	**4.886**	**1.096**
	std	**15.320**	1.213	**0.220**	15.397	**0.944**	0.227

TABLE IV
COMPARISON BETWEEN BASIC BAGGING AND LAYERED BAGGING BASED ON WIN-LOSS COUNT FOR THE TIME SERIES DATA OF THE NN3 FORECASTING COMPETITION [37]. HERE, THE BEST RESULT IS HIGHLIGHTED USING BOLDFACE TEXT

Performance Metrics	Number of Wins	
	Basic Bagging	Layered Bagging
sMAPE	26	**85**
MASE	32	**79**
MdRAE	25	**86**

Tables II–IV and Fig. 2 show the results of basic bagging and layered bagging. Several observations can be made from these results.

1) It can be seen that the lag obtained by the first ensemble layer of LEA is different for different time series (Table II). These results indicate that it is very important to determine the lag automatically. Note that the basic bagging algorithm uses the same lag for different time series. According to [37], the lag is set to 12 for basic bagging.
2) In terms of average sMAPE, MASE, and MdRAE layered bagging is found superior than basic bagging

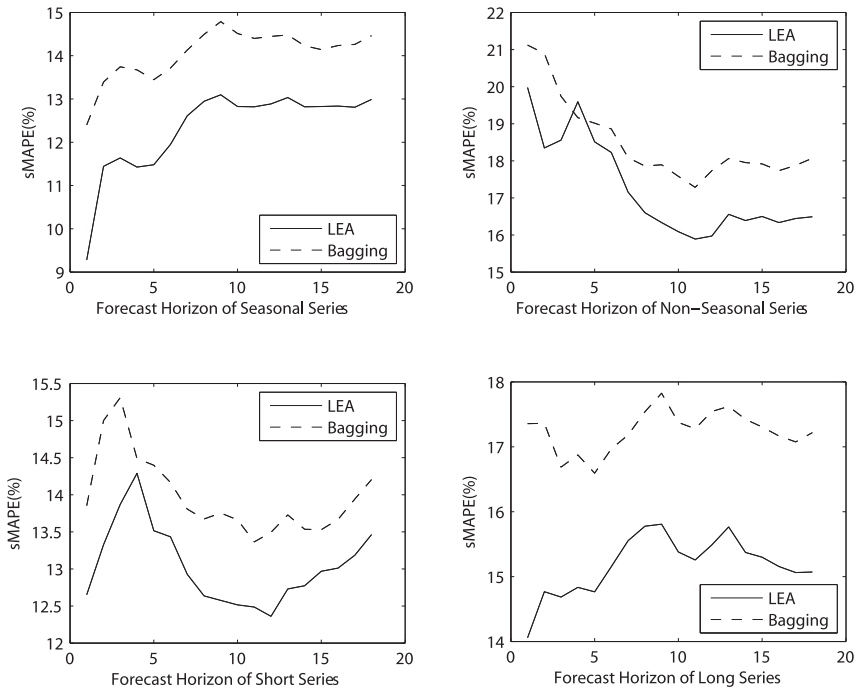

Fig. 2. Comparison between basic bagging and layered bagging for different forecast horizons based on the average sMAPE of seasonal, nonseasonal, short and long time series data of the NN3 forecasting competition [37].

irrespective of the nature of time series. For example, for the seasonal time series, the average MdRAE achieved by basic bagging is 0.618, whereas it achieved by layered bagging is 0.546 (Table III).

3) The performance of layered bagging is found more consistent compared to basic bagging across different time series. This can be observed by looking at the standard deviation of these two methods (Table III).
4) It can be observed from Fig. 2 that layered bagging is better than basic bagging for different forecast horizons. It is also observed that in several occasions the average sMAPE of layered bagging and basic bagging increases with increasing forecast horizons. Although the effect of forecast horizon presented in Fig. 2 is based on average sMAPE, a very similar effect can be observed based on MASE or MdRAE.
5) To get an idea about the performance on different time series, we count the number of times layered bagging is better or worse than basic bagging. For any performance measurement, if layered bagging exhibits better performance than basic bagging for a particular time series, we consider it as win for layered bagging; otherwise it is a loss. Once again, layered bagging is proved superior than basic bagging with respect to three different accuracy measures (Table IV).

Following the suggestion from [58] and [59] and to facilitate a fair comparison (see [14], [28], [38], [60]), we use the Wilcoxon signed rank test to assess whether the performance difference between two methods is statistically significant. Compared to the *t*-test, the ranked test makes fewer and less stringent assumptions on the sample distributions and thus is more powerful in detecting the existence of significant differences. In our test, the matched pair samples are the per-series sMAPE, MASE, and MdRAE obtained for the forecasting horizon 18.

TABLE V
WILCOXON SIGNED RANK TEST SUMMARY BETWEEN LAYERED BAGGING AND BASIC BAGGING FOR THE TIME SERIES DATA OF THE NN3 FORECASTING COMPETITION [37]

	Performance Metrics	R^+	R^-	p-value	Null hypothesis Significance level=0.05
Seasonal	sMAPE	1899	447	9.16E-06	Rejected for layered bagging
	MASE	1802	544	1.20E-04	Rejected for layered bagging
	MdRAE	1867	479	2.20E-05	Rejected for layered bagging
Non-seasonal	sMAPE	819	127	2.94E-05	Rejected for layered bagging
	MASE	749	197	8.60E-03	Rejected for layered bagging
	MdRAE	787	159	1.60E-04	Rejected for layered bagging
Short	sMAPE	1068	207	3.24E-05	Rejected for layered bagging
	MASE	930	345	4.80E-04	Rejected for layered bagging
	MdRAE	1085	190	1.56E-05	Rejected for layered bagging
Long	sMAPE	1564	327	8.89E-06	Rejected for layered bagging
	MASE	1560	331	1.02E-05	Rejected for layered bagging
	MdRAE	1512	379	4.72E-05	Rejected for layered bagging

Table V shows the summary of the Wilcoxon signed rank test based on the sMAPE, MASE, and MdRAE. Here, R^+ corresponds to the sum of ranks for layered bagging and R^- for basic bagging. These notations are used throughout this paper. The results show that the null hypothesis has been rejected in favor of layered bagging with a significance level 0.05 for four different time series we compare here. In fact, the associated *p*-values indicate that even a significance level 0.001 would have resulted in the rejection.

We now analyze the ensembles produced by layered bagging and basic bagging using bias-variance-covariance decomposition [35], [61], double fault [62] and disagreement [62].

a) Bias-variance-covariance estimation: Mean-square-error, E_{mse}, of an ensemble can be decomposed into bias (E_{bias}), variance (E_{var}), and covariance (E_{cov}). For regression problems, this decomposition has been widely used (see [35], [61]) for analyzing the performance of ensembles and can be expressed as

$$E_{\text{mse}} = E_{\text{bias}} + E_{\text{var}} + E_{\text{cov}}. \tag{8}$$

The above equation indicates that the bias, variance, and covariance should be small for achieving a good performance by an ensemble.

Let the average outputs of the ensemble and network i on the nth pattern in the testing set are denoted by $\overline{\hat{Y}}(n)$ and $\overline{\hat{Y}}_i(n)$, respectively. These outputs can be expressed as

$$\overline{\hat{Y}}(n) = \frac{1}{K}\sum_{k=1}^{K} \hat{Y}^{(k)}(n) \tag{9}$$

$$\overline{\hat{Y}}_i(n) = \frac{1}{K}\sum_{k=1}^{K} \hat{Y}_i^{(k)}(n) \tag{10}$$

where $\hat{Y}^{(k)}(n)$ and $\hat{Y}_i^{(k)}(n)$ are the output of the ensemble and the network i on the nth pattern from the kth simulation. The bias, variance and covariance of the ensemble containing M networks can be defined in the following manner:

$$E_{\text{bias}} \equiv \frac{1}{N}\sum_{n=1}^{N}\left(\overline{\hat{Y}}(n) - Y(n)\right)^2 \tag{11}$$

where $Y(n)$ is the actual output of the nth pattern and N is the total number of patterns in the testing set

$$E_{\text{var}} \equiv \sum_{i=1}^{M}\frac{1}{N}\sum_{n=1}^{N}\frac{1}{K}\sum_{k=1}^{K}\frac{1}{M^2}\left(\hat{Y}_i^{(k)}(n)\right) - \overline{\hat{Y}}_i\,(n))^2 \tag{12}$$

$$E_{\text{cov}} \equiv \sum_{i=1}^{M}\sum_{j=1,j\neq i}^{M}\frac{1}{N}\sum_{n=1}^{N}\frac{1}{K}\sum_{k=1}^{K}\frac{1}{M^2}\left(\hat{Y}_i^{(k)}(n)\right) - \overline{\hat{Y}}_i(n))^2\left(\hat{Y}_j^{(k)}(n)\right) - \overline{\hat{Y}}_j\,(n))^2. \tag{13}$$

E_{mse} can also be defined by

$$E_{\text{mse}} \equiv \frac{1}{N}\sum_{n=1}^{N}\frac{1}{K}\sum_{k=1}^{K}\left(\hat{Y}^{(k)}(n) - Y(n)\right)^2. \tag{14}$$

To obtain the bias, variance, and covariance of the ensemble, we follow the experimental methodology suggested in [61]. According to [61], several (say, 25) simulations of the ensemble have to be conducted. The only difference in different simulations is the data sets used for training the networks. Since the NN3 competition [37] contains a large number of time series, we select only one series for each of four different categories i.e., the series numbers 2, 71, 73, and 110 for short, seasonal, nonseasonal, and long categories, respectively. Note that a similar result can be obtained for other series.

Table VI summarizes the average results of the bias-variance-covariance decomposition obtained for layered bagging and basic bagging. It can be observed from this table that layered bagging provides less bias than basic bagging.

TABLE VI
COMPARISON BETWEEN LAYERED BAGGING AND BASIC BAGGING BASED ON AVERAGE BIAS, VARIANCE AND COVARIANCE FOR THE TIME SERIES 2, 71, 73, AND 110 OF THE NN3 FORECASTING COMPETITION [37]. NOTE THAT THE SERIES NUMBERS 2, 71, 73, AND 110 REPRESENT THE SHORT, SEASONAL, NONSEASONAL, AND LONG TIME SERIES, RESPECTIVELY. HERE, THE BEST RESULT IS HIGHLIGHTED USING BOLDFACE TEXT

		E_{bias}	E_{var}	E_{cov}	E_{mse}
Layered Bagging	Seasonal	**0.0705**	**4.21E-04**	**0.0101**	**0.0810**
	Non-seasonal	**0.0513**	**4.40E-04**	**0.0055**	**0.0572**
	Short	**0.0538**	**8.55E-04**	**0.0317**	**0.0863**
	Long	**0.0419**	7.87E-04	**0.0276**	**0.0702**
Basic Bagging	Seasonal	0.1026	4.63E-04	0.0046	0.1076
	Non-seasonal	0.0874	3.00E-03	0.0376	0.1280
	Short	0.0735	1.20E-03	0.0476	0.1223
	Long	0.1157	**6.78E-04**	0.0112	0.1275

Once again, the effectiveness of using the appropriate lag obtained by the first ensemble layer of layered bagging is evident. Layered bagging also produces less variance and covariance than basic bagging for most of the cases. The positive effect of less bias, variance, and covariance is the less E_{mse}, as shown in the last column of Table VI.

b) Disagreement and double fault: Although many diversity measures have been proposed in the context of classification problems, few such measures have been proposed for regression problems. We here analyze layered bagging and basic bagging based on two diversity measures: disagreement and double fault. Disagreement is the ratio between the number of observations on which one network is correct and the other network is incorrect to the total number of observations. Double fault is defined as the portion of the observations for which the two networks produce wrong answers.

To use disagreement and double fault measures for TSF problems, we use the extension suggested in [62]. For each instance X, we calculate the standard deviation, σ, of the estimated target variable by all networks in an ensemble. If the true value of the target is α, then a prediction β is considered to be correct if $\beta \leq \alpha + \sigma$ and $\beta \geq \alpha - \sigma$ i.e., the prediction has to fall within a margin of one standard deviation of the value of the target variable. Otherwise, the prediction is considered as incorrect.

Table VII summarizes the average results of disagreement and double fault for basic bagging and layered bagging for 111 time series of NN3. It can be observed that in terms of double fault, layered bagging is found better than basic bagging irrespective of the nature of the time series. As layered bagging tries to enforce accuracy among the members of an ensemble, it is obvious that the number of instances for which a pair of networks makes mistake will be less for layered bagging. This is the main reason for obtaining the better double fault measure by our method. In terms of the disagreement measure, both basic bagging and layered bagging are better for two cases out of four different time series.

2) Comparison With Boosting: At each iteration, boosting reweighs training examples in such a way that the incorrectly classified examples get more weights. Hence, we do not need to specify any data partition percentage here like bagging.

TABLE VII
COMPARISON BETWEEN BASIC BAGGING AND LAYERED BAGGING BASED ON AVERAGE DISAGREEMENT AND DOUBLE FAULT FOR SEASONAL, NONSEASONAL, SHORT AND LONG TIME SERIES DATA OF THE NN3 FORECASTING COMPETITION [37]. HERE, THE BEST RESULT IS HIGHLIGHTED USING BOLDFACE TEXT

	Basic Bagging		Layered Bagging	
	disagree	double fault	disagree	double fault
Seasonal	0.255	0.393	**0.271**	**0.365**
Non-seasonal	**0.352**	0.342	0.321	**0.319**
Short	**0.370**	0.390	0.327	**0.361**
Long	0.228	0.359	**0.260**	**0.335**

To incorporate layering, we simply implement two layers of boosting and named this version as layered boosting. Layered boosting also outperforms basic boosting in terms of three error measures we use in this paper. We also use here win-loss count to compare these two versions of boosting, which also provides superiority of our layered boosting. The Wilcoxon signed rank test conducted based on three error measures also confirms that the performance of layered boosting is significantly better than its counterpart. The details of these comparisons are presented in the supplementary materials.

3) Comparison With Naïve Forecast: In the TSF (see [63], [64]), the naïve forecast is used as a benchmark model against which more sophisticated models can be compared. This benchmark model assumes that the forecast of any data point, d_i, equals to the actual value of the d_{i-1} point. The Wilcoxon signed rank test indicates that our layered bagging and layered boosting significantly outperform the naïve forecast model. The detail comparisons are given in the supplementary materials.

4) Comparison With Other Work: The NN3 forecasting competition attracts 59 submissions from CI-based and statistical techniques-based methods, making this as the largest CI competition on time series. We choose the best five benchmark statistical methods and the best five CI-based methods for our comparison. The detail description of these methods can be found in [60]. Furthermore, we choose a recently introduced CI-based method [28], which employ an ensemble of RBF networks, for comparison.

Table VIII presents the average results over 111 time series of NN3. The method Identifications (IDs) with letter C as prefix and those with letter B in this table stand for CI-based and statistical benchmark-based methods, respectively. It can be observed that our layered bagging beats not only CI-based methods but also the benchmark statistical methods in terms of average sMAPE, MASE, and MdRAE. This comparison indicates that the layered ensemble approach with proper techniques for maintaining accuracy and diversity is useful for obtaining a good forecasting accuracy. It is here worth mentioning that the best CI-based method (see [28]) and LEA employ the same preprocessing techniques.

TABLE VIII
COMPARISON AMONG LAYERED BAGGING, YAN [28] AND TEN OTHER METHODS [60] BASED ON AVERAGE SMAPE, MASE, AND MDRAE. NOTE THAT THE RESULTS ARE AVERAGE OF 111 TIME SERIES DATA OF THE NN3 FORECASTING COMPETITION [37] AND "-" REPRESENTS THE RESULT IS NOT AVAILABLE. HERE, THE BEST RESULT IS HIGHLIGHTED USING BOLDFACE TEXT

ID	Method	sMAPE	MdRAE	MASE
-	**Layered bagging**	**14.56**	**0.55**	**1.02**
B09	Wildi	14.84	0.82	1.13
B07	Theta	14.89	0.88	1.13
C27	Illies	15.18	0.84	1.25
B03	ForecastPro	15.44	0.89	1.17
-	Yan	15.80	-	-
B16	DES	15.90	0.94	1.17
B17	Comb S-H-D	15.93	0.90	1.21
C03	Flores	16.13	0.93	1.20
C46	Chen	16.55	0.94	1.34
C13	D'yakonov	16.57	0.91	1.26
C50	Kamel	16.92	0.90	1.28

D. Further Experiments and Comparison

We further evaluate the performance of layered bagging on the NN5 time series [38], another large benchmark time series datasets. The difficulties of these datasets include outliers, missing values, multiple overlying seasonality, etc. All these make the NN5 forecasting competition is one of the most interesting competition. Like the NN3 forecasting competition, the NN5 forecasting competition provides two datasets A and B. The dataset A contains 111 daily time series representing roughly two years of daily cash money withdrawal (735 data points) from ATM machines at various cities in the U.K. The dataset B, on the other hand, contains only 11 series taken from the dataset A. The main challenge of the NN5 forecasting competition is to forecast the values of the next 56 days (i.e., $h = 56$). This competition utilizes sMAPE to measure the performance of different methods. It is worth mentioning that, we here use the same experimental configuration of the NN3 competition except the ensemble size, which is set to 100.

The average sMAPE over 111 time series of layered bagging and ten other algorithms of the NN5 forecasting competition is furnished in Table IX. Note that the results of the other algorithms are collected from [38]. Like the NN3 competition, the method IDs with letter C as prefix and those with letter B stand for CI-based methods and statistical benchmark methods, respectively. It can be observed that our layered bagging again beats all the CI and statistical benchmark-based methods (Table IX).

Finally, we compare our layered bagging with standard bagging and ensemble approaches described in [65] on several traditional time series such as Henon, Ikeda, Sunspot, SP 500, and Dow Jones. We here use here the same experimental setup and performance metric as [65]. Comparison is done based on normalized root mean square error (NRMSE), which can be defined as

$$\text{NRMSE} = \frac{\sum_{n=1}^{N}\left(\hat{Y}(n) - Y(n)\right)^2}{\sum_{n=1}^{N}\left(\hat{Y}(n) - \bar{Y}\right)^2} \tag{15}$$

where $\bar{Y}$ represents the mean of the actual outputs.

The results of bagging, layered bagging, ensemble (average) [65], and ensemble (rank-based linear combination (RBLC)) [65] are summarized in Table X. Our layered

TABLE IX
COMPARISON AMONG LAYERED BAGGING AND 10 OTHER METHODS [38] BASED ON AVERAGE sMAPE. NOTE THAT THE RESULTS ARE AVERAGE OF 111 TIME SERIES DATA OF NN5 [38] COMPETITION AND - REPRESENTS DATA ARE NOT AVAILABLE. HERE, THE BEST RESULT IS HIGHLIGHTED USING BOLDFACE TEXT

ID	Method	sMAPE
-	**Layered bagging**	**19.88**
B02	Wildi	19.9
C23	Andrawis	20.4
C12	Vogel	20.5
C10	D'yakonov	20.6
B08	Noncheva	21.1
C06	Rauch	21.7
C19	Luna	21.8
B05	Lagoo	21.9
C01	Wichard	22.1
C17	Gao	22.3

TABLE X
COMPARISON BETWEEN BASIC BAGGING, LAYERED BAGGING, ENSEMBLE (AVERAGE) [65], AND ENSEMBLE (RBLC) [65] BASED ON AVERAGE NRMSE FOR FIVE DIFFERENT TRADITIONAL TIME SERIES. HERE, THE BEST RESULT IS HIGHLIGHTED USING BOLDFACE TEXT

	Basic Bagging	Layered Bagging	Ensemble (Average)	Ensemble (RBLC)
Henon	0.1001	**0.0798**	0.1078	0.0938
Ikeda	0.2956	**0.2762**	0.3224	0.3097
Sunspot	0.0056	**0.0046**	0.0063	0.0057
SP 500	0.0350	**0.0227**	0.0374	0.0355
Dow Jones	0.0270	**0.0183**	0.0277	0.0267

bagging beats not only traditional bagging but also ensemble (average) and ensemble (RBLC). These results clearly indicate the superiority of our layered approach for obtaining a good forecasting performance.

V. SENSITIVITY

This section presents the sensitivity of data preprocessing, ensemble size, and data resampling on LEA. We use 111 time series data of the NN3 forecasting competition [37] for our sensitivity analysis.

A. Effect of Data Preprocessing

In this paper, the data preprocessing step consists of noise removal, missing value treatment, and handling seasonality (called deseasonalization). We implement layered bagging and basic bagging: with noise removal and without noise removal, with missing value treatment and without missing value treatment, with deseasonalization and without deseasonalization. To show the effect of these components on forecasting accuracy, we apply the analysis of variance (ANOVA) test on the results obtained by layered bagging and basic bagging.

Table XI shows the ANOVA test results. It can be seen from this table that with a p-value greater than 0.05, both layered bagging and basic bagging are generally not sensitive to noise removal and treatment of missing value. However, the seasonal adjustment has a significant impact on the forecasting accuracy of layered bagging and basic bagging, with a p-value of 0.0009 and 0.0017, respectively. The results presented in this section indicate that data processing improves the performance of layered bagging and basic bagging. Since the sum of square errors of basic bagging is higher than layered bagging (Table XI), basic bagging is more sensitive to data preprocessing than layered bagging. It is now understood that preprocessing does not provide any extra favor to our approach.

TABLE XI
ANOVA TEST RESULTS FOR THE SENSITIVITY OF NOISE REMOVAL, MISSING VALUE TREATMENT, AND HANDLING SEASONALITY (DESEASONALIZATION)

method	parameter	Degree of Freedom	Sum of Squares Error	F-value	P-value
Layered Bagging	Noise Removal	1	16.400	0.090	0.7690
	Missing Value	1	102.60	0.580	0.4450
	Deseasonalization	1	3560.5	11.28	0.0009
Basic Bagging	Noise Removal	1	23.400	0.08	0.7715
	Missing Value	1	251.60	0.97	0.3252
	Deseasonalization	1	4272.5	12.50	0.0017

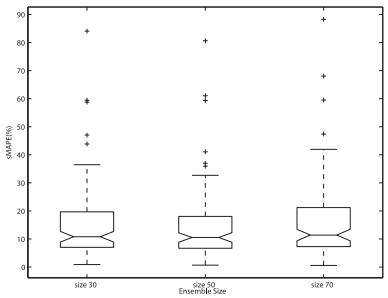

Fig. 3. Effect of the ensemble size on the performance of layered bagging based on the average sMAPE of the 111 time series data of the NN3 forecasting competition [37].

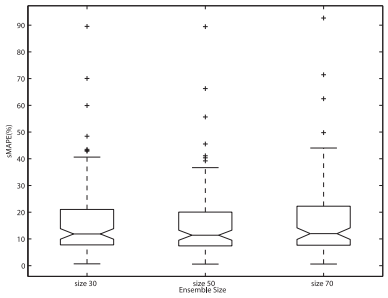

Fig. 4. Effect of the ensemble size on the performance of basic bagging based on the average sMAPE of the 111 time series data of the NN3 forecasting competition [37].

B. Effect of Ensemble Size

We choose the ensemble size τ (i.e., the number of MLP networks) manually (see [32], [43]). According to [43], any value between 50 and 100 is suitable for τ, which is common in the ensemble literature. We choose τ equal to 50 in our experiments related to the time series data of the NN3 forecasting competition. We here vary τ from 30 to 70 to show the effect of the ensemble size on layered bagging and basic bagging.

Figs. 3 and 4 illustrate the average sMAPE obtained by layered bagging and basic bagging for different ensemble sizes.

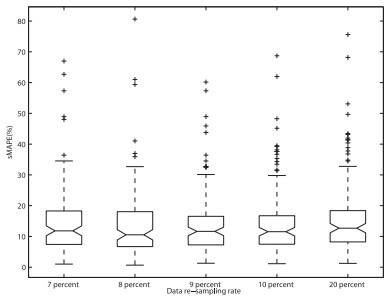

Fig. 5. Effect of the data resampling rate on the performance of layered bagging based on the average sMAPE of the 111 time series data of the NN3 forecasting competition [37].

It is evident from these figures that the performance of layered bagging and basic bagging increases when the ensemble size increases up to a certain limit. For example, when we increase the ensemble size from 30 to 50, the average sMAPE of layered bagging and basic bagging reduces from 15.05% to 14.56% and from 16.47% to 15.93%, respectively. However, increasing the ensemble size beyond 50 is not enhancing the performance rather giving inferior results in some cases. We again see here the ensemble size affects the performance of layered bagging and basic bagging in the same way.

C. Effect of Resampling Rate and Resampling

In LEA, we try to encourage diversity among the networks of the second ensemble layer by applying a resampling technique to create different training sets. We need to assign the resampling rate when LEA uses bagging as the resampling technique. However, it is not required when boosting is applied as a resampling technique. In this section, we investigate whether the resampling rate affects LEA. We also investigate whether the use of the resampling technique is beneficial.

In second ensemble layer of LEA, we apply random resampling on the optimal dataset D_{tr} to obtain different training sets. Unlike basic bagging which uses a fixed data resampling rate to create different training sets, we resample $k\%$ data, a user defined parameter, for the same purpose. Fig. 5 illustrates the results of layered bagging for different data resampling rates. It is evident from this figure that increasing the data resampling rate is beneficiary for layered bagging, but only up to a certain limit. For example, the average sMAPE of layered bagging reduces from 15.26% to 14.56% if we increase the data resample rate from 7% to 9%. But after that increasing the data resample rate decreases the performance of layered bagging (Fig. 5). We can say that determining an optimal resampling rate is important for achieving a good forecasting accuracy.

Is it really necessary to apply data resampling to improve the generalization performance of LEA. To investigate this issue, we run LEA with resampling (bagging) and without bagging. The value of data resampling rate k is set to 9% for LEA with resampling. Table XII illustrates the results of our experiment. From this table, we can say that the use of data resampling really improves both accuracy and diversity of the networks used for obtaining the ensemble output of the second ensemble layer.

TABLE XII
EFFECT OF RESAMPLING BASED ON AVERAGE sMAPE, MASE, MDRAE, DOUBLE FAULT, AND DISAGREEMENT OF LEA. NOTE THAT THE RESULTS ARE AVERAGE OF 111 TIME SERIES DATA OF THE NN3 FORECASTING COMPETITION [37]. HERE, THE BEST RESULT IS HIGHLIGHTED USING BOLDFACE TEXT

Performance Metrics	LEA with bagging	LEA without bagging
sMAPE	**14.56**	18.73
MASE	**1.02**	1.24
MdRAE	**0.55**	0.67
double fault	**0.34**	0.41
disagreement	**0.29**	0.21

VI. CONCLUSION

Ensembles have been introduced to the machine learning community for several decades. Although many ensemble algorithms have been developed for classification problems, few such algorithms exist for dealing with TSF problems. An important component of ensembles algorithms developed either for TSF or classification problems is the consideration of accuracy and diversity among the members that constitute an ensemble. Existing ensemble algorithms (see [14], [18], [19], [20]) consider only accuracy or diversity but not both for solving TSF problems. In this paper, we propose LEA, a LEA, for accurately forecasting time series data. Our layered architecture is consisted of two layers, each of which is an ensemble of neural networks. Accuracy of the base predictors used for forecasting is ensured by employing the appropriate lag, while diversity is encouraged by training the predictors on different training sets. Furthermore, LEA does not combine all base predictors rather a subset of predictors that are accurate and diverse.

Extensive experiments have been carried out in this paper to evaluate how well LEA performs on different TSF problems in comparison with other ensemble and nonensemble algorithms. In almost all cases, LEA has been found better compared to popular ensemble algorithms bagging and boosting. These results indicate that the layering concept we introduce in this paper can help to improve the forecasting accuracy of basic ensemble algorithms irrespective of the type of time series. When we compare LEA to other state-of-the-art statistical and CI-based methods, our algorithm has also been found better in this case. In its current implementation, LEA uses MLP networks as base predictors, other types of networks such as RBF networks and recurrent neural networks can be used as base predictors in future. We use a fixed ensemble size and a same k value for data resampling irrespective of the type and complexity of time series data. One of the future improvements to LEA would be to make them adaptive.

REFERENCES

[1] D. N. P. Murthy and K. C. Staib, "Forecasting maximum speed of aeroplanes—A case study in technology forecasting," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. SMC-14, no. 2, pp. 304–310, Mar./Apr. 1984.

[2] S.-M. Chen and J.-R. Hwang, "Temperature prediction using fuzzy time series," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 30, no. 2, pp. 263–265, Apr. 2000.

[3] K. Huarng and T. H.-K. Yu, "Ratio-based lengths of intervals to improve fuzzy time series forecasting," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 36, no. 2, pp. 328–340, Apr. 2006.

[4] T. H. Naylor, T. G. Seaks, and D. W. Wichern, "Box-Jenkins methods: An alternative to econometric models," *Int. Stat. Rev.*, vol. 40, no. 2, pp. 123–137, 1972.

[5] H.-S. Yan and X. Tu, "Short-term sales forecasting with change-point evaluation and pattern matching algorithms," *Expert Syst. Appl.*, vol. 39, no. 5, pp. 5426–5439, 2012.

[6] H.-S. Yan, Q. Wu, and X. Tu, "A short-term forecasting model with inhibiting normal distribution noise of sale series," *Appl. Artif. Intell.*, vol. 27, no. 6, pp. 496–519, 2013.

[7] P. Cortez, M. Rocha, F. S. Allegro, and J. Neves, "Real-time forecasting by bio-inspired models," in *Proc. 2nd Int. Conf. Artif. Intell. Appl.*, Málaga, Spain, 2002, pp. 52–57.

[8] P. Cortez, M. Rocha, and J. Neves, "Evolving time series forecasting ARMA models," *J. Heuristics*, vol. 10, no. 4, pp. 415–429, 2004.

[9] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Netw.*, vol. 2, no. 5, pp. 359–366, 1989.

[10] L. I. Kuncheva, "Switching between selection and fusion in combining classifiers: An experiment," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 32, no. 2, pp. 146–156, Apr. 2002.

[11] T. G. Dietterich, "Ensemble methods in machine learning," in *Multiple Classifier Systems*. Berlin, Germany: Springer, 2000, pp. 1–15.

[12] G. Brown, J. Wyatt, R. Harris, and X. Yao, "Diversity creation methods: A survey and categorization," *Inf. Fusion*, vol. 6, no. 1, pp. 5–20, 2005.

[13] R. T. Clemen, "Combining forecasts: A review and annotated bibliography," *Int. J. Forecasting*, vol. 5, no. 4, pp. 559–583, 1989.

[14] H. Chen and X. Yao, "Ensemble regression trees for time series predictions," *Neurocomputing*, vol. 70, nos. 4–6, pp. 697–703, 2007.

[15] G. Zhang and V. Berardi, "Time series forecasting with neural network ensembles: An application for exchange rate prediction," *J. Oper. Res. Soc.*, vol. 52, no. 6, pp. 652–664, 2001.

[16] R. Schapire, "The strength of weak learnability," *Mach. Learn.*, vol. 5, no. 2, pp. 197–227, 1990.

[17] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, 1996.

[18] R. Avnimelech and N. Intrator, "Boosting regression estimators," *Neural Comput.*, vol. 11, no. 2, pp. 499–520, 1999.

[19] M. Assaad, R. Boné, and H. Cardot, "A new boosting algorithm for improved time-series forecasting with recurrent neural networks," *Inf. Fusion*, vol. 9, no. 1, pp. 41–55, 2008.

[20] W. Goh, C. Lim, and K. Peh, "Predicting drug dissolution profiles with an ensemble of boosted neural networks: A time series approach," *IEEE Trans. Neural Netw.*, vol. 14, no. 2, pp. 459–463, Mar. 2003.

[21] G. Paris, D. Robilliard, and C. Fonlupt, "Applying boosting techniques to genetic programming," in *Artificial Evolution*. Berlin, Germany: Springer, 2002, pp. 267–278.

[22] L. V. de Souza, A. T. Pozo, J. M. da Rosa, and A. C. Neto, "The boosting technique using correlation coefficient to improve time series forecasting accuracy," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Singapore, 2007, pp. 1288–1295.

[23] Z. Zheng, "Boosting and bagging of neural networks with applications to financial time series," M.S. thesis, Dept. Stat., Univ. Chicago, Chicago, IL, USA, 2006.

[24] S. Kurogi, R. Koyama, S. Tanaka, and T. Sanuki, "Forecasting using first-order difference of time series and bagging of competitive associative nets," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN)*, Orlando, FL, USA, 2007, pp. 166–171.

[25] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[26] I. Ilies *et al.*, "Stepping forward through echoes of the past: Forecasting with echo state networks," *Short Report Winning Entry NN3 Financ. Forecasting Compet. 2007*, Jun. 2014, [Online]. Available: http://www:neural-forecasting-competition.com/

[27] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural Comput.*, vol. 1, no. 2, pp. 270–280, Jun. 1989.

[28] W. Yan, "Toward automatic time-series forecasting using neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 7, pp. 1028–1039, Jul. 2012.

[29] J. Wichard and M. Ogorzalek, "Time series prediction with ensemble models," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, vol. 2. Budapest, Hungary, 2004, pp. 1625–1630.

[30] D. H. Wolpert, "The lack of a priori distinctions between learning algorithms," *Neural Comput.*, vol. 8, no. 7, pp. 1341–1390, Oct. 1996.

[31] D. Ruta, B. Gabrys, and C. Lemke, "A generic multilevel architecture for time series prediction," *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 3, pp. 350–359, Mar. 2011.

[32] D. K. Barrow, S. F. Crone, and N. Kourentzes, "An evaluation of neural network ensembles and model selection for time series prediction," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Barcelona, Spain, 2010, pp. 1–8.

[33] M. M. Islam, X. Yao, S. S. Nirjon, M. A. Islam, and K. Murase, "Bagging and boosting negatively correlated neural networks," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 38, no. 3, pp. 771–784, Jun. 2008.

[34] A. J. Sharkey and N. E. Sharkey, "Combining diverse neural nets," *Knowl. Eng. Rev.*, vol. 12, no. 3, pp. 231–247, 1997.

[35] Y. Liu and X. Yao, "Simultaneous training of negatively correlated neural networks in an ensemble," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 29, no. 6, pp. 716–725, Dec. 1999.

[36] D. W. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," *J. Soc. Ind. Appl. Math.*, vol. 11, no. 2, pp. 431–441, 1963.

[37] (Jun. 20, 2014). *Time Series Forecasting Competition for Neural Networks and Computational Intelligence*. [Online]. Available: http://www.neural-forecasting-competition.com/NN3

[38] (Jun. 20, 2014). *Time Series Forecasting Competition for Neural Networks and Computational Intelligence*. [Online]. Available: http://www.neural-forecasting-competition.com/NN5

[39] M. Adya, F. Collopy, J. Armstrong, and M. Kennedy, "Automatic identification of time series features for rule-based forecasting," *Int. J. Forecasting*, vol. 17, no. 2, pp. 143–157, 2001.

[40] J. D. Wichard, "Forecasting the NN5 time series with hybrid models," *Int. J. Forecasting*, vol. 27, no. 3, pp. 700–707, 2011.

[41] R. R. Andrawis, A. F. Atiya, and H. El-Shishiny, "Forecast combinations of computational intelligence and linear models for the NN5 time series forecasting competition," *Int. J. Forecasting*, vol. 27, no. 3, pp. 672–688, 2011.

[42] M. Adya and F. Collopy, "How effective are neural networks at forecasting and prediction? A review and evaluation," *J. Forecasting*, vol. 17, nos. 5–6, pp. 481–495, 1998.

[43] M. Ponti, "Combining classifiers: From the creation of ensembles to the decision fusion," in *Proc. 24th SIBGRAPI Conf. Graph. Patterns Images Tuts. (SIBGRAPI-T)*, Alagoas, Brazil, 2011, pp. 1–10.

[44] G. Zhang, "Linear and nonlinear time series forecasting with artificial neural networks," Ph.D. dissertation, Graduate School Manage., Kent State Univ., Kent, OH, USA, 1998.

[45] D. Ruta and B. Gabrys, "Classifier selection for majority voting," *Inf. Fusion*, vol. 6, no. 1, pp. 63–81, 2005.

[46] G. Giacinto and F. Roli, "An approach to the automatic design of multiple classifier systems," *Pattern Recognit. Lett.*, vol. 22, no. 1, pp. 25–33, 2001.

[47] G. Giacinto and F. Roli, "Design of effective neural network ensembles for image classification purposes," *Image Vis. Comput.*, vol. 19, no. 9, pp. 699–707, 2001.

[48] A. Krogh, J. Vedelsby, and T. K. Leen, "Neural network ensembles, cross validation, and active learning," in *Advances in Neural Information Processing Systems*. Denver, CO, USA: MIT Press, 1995, pp. 231–238.

[49] S. Hashem, "Optimal linear combinations of neural networks," *Neural Netw.*, vol. 10, no. 4, pp. 599–614, 1997.

[50] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Trans. Evol. Comput.*, vol. 3, no. 2, pp. 82–102, Jul. 1999.

[51] L. K. Hansen and P. Salamon, "Neural network ensembles," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 10, pp. 993–1001, Oct. 1990.

[52] M. M. Islam, X. Yao, and K. Murase, "A constructive algorithm for training cooperative neural network ensembles," *IEEE Trans. Neural Netw.*, vol. 14, no. 4, pp. 820–834, Jul. 2003.

[53] R. R. Andrawis and A. F. Atiya, "A new Bayesian formulation for Holt's exponential smoothing," *J. Forecasting*, vol. 28, no. 3, pp. 218–234, 2009.

[54] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *Int. J. Forecasting*, vol. 22, no. 4, pp. 679–688, 2006.

[55] (Jun. 20, 2014). *Time Series Forecasting Competition for Neural Networks and Computational Intelligence*. [Online]. Available: http://www.neural-forecasting-competition.com/index.htm

[56] J. S. Armstrong and F. Collopy, "Error measures for generalizing about forecasting methods: Empirical comparisons," *Int. J. Forecasting*, vol. 8, no. 1, pp. 69–80, 1992.

[57] J. Mager, U. Paasche, and B. Sick, "Forecasting financial time series with support vector machines based on dynamic kernels," in *Proc. IEEE Conf. Soft Comput. Ind. Appl. (SMCIA)*, Muroran, Japan, 2008, pp. 252–257.

[58] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, and F. Herrera, "An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes," *Pattern Recognit.*, vol. 44, no. 8, pp. 1761–1776, 2011.

[59] A. Fernández, S. García, J. Luengo, E. Bernadó-Mansilla, and F. Herrera, "Genetics-based machine learning for rule induction: State of the art, taxonomy, and comparative study," *IEEE Trans. Evol. Comput.*, vol. 14, no. 6, pp. 913–941, Dec. 2010.
[60] S. F. Crone, M. Hibon, and K. Nikolopoulos, "Advances in forecasting with neural networks? Empirical evidence from the NN3 competition on time series prediction," *Int. J. Forecasting*, vol. 27, no. 3, pp. 635–660, 2011.
[61] R. A. Jacobs, "Bias/variance analyses of mixtures-of-experts architectures," *Neural Comput.*, vol. 9, no. 2, pp. 369–383, 1997.
[62] H. Dutta, "Measuring diversity in regression ensembles," in *Proc. 4th Indian Int. Conf. Artif. Intell. (IICAI)*, vol. 9. Karnataka, India, 2009, pp. 2220–2236.
[63] A. Timmermann, G. Elliott, and C. W. J. Granger, "Forecast combinations," in *Handbook of Economic Forecasting*, vol. 1. Amsterdam, The Netherlands: Elsevier, 2006, pp. 135–196.
[64] J. S. Armstrong, *Principles of Forecasting: A Handbook for Researchers and Practitioners*, vol. 30. New York, NY, USA: Springer, 2001.
[65] V. M. Landassuri-Moreno and J. A. Bullinaria, "Neural network ensembles for time series forecasting," in *Proc. Genet. Evol. Comput. Conf. (GECCO)*, Montreal, QC, Canada, 2009, pp. 1235–1242.
[66] A. Inoue and L. Kilian, "In-sample or out-of-sample tests of predictability: Which one should we use?" *Econ. Rev.*, vol. 23, no. 4, pp. 371–402, 2005.
[67] J. Wichard and M. Ogorzalek, "Time series prediction with ensemble models," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, vol. 2. Budapest, Hungary, 2004, pp. 1625–1630.

Md. Mustafizur Rahman received the B.Sc. and M.Sc. engineering degrees in computer science and engineering from the Bangladesh University of Engineering and Technology, Dhaka, Bangladesh, in 2011 and 2013, respectively. He is currently pursuing the Ph.D. degree from the Department of Computer Science, University of Virginia, Charlottesville, VA, USA.

His current research interests include machine learning and its applications to user behavior modeling, natural language processing, text data analysis, and probabilistic graphical model.

Md. Monirul Islam (M'10) received the B.E. degree from the Khulna University of Engineering and Technology (KUET), Khulna, Bangladesh, the M.E. degree from the Bangladesh University of Engineering and Technology (BUET), Dhaka, Bangladesh, and the Ph.D. degree from the University of Fukui, Fukui, Japan, in 1989, 1996, and 2002, respectively.

From 1989 to 2002, he was a Lecturer and an Assistant Professor with KUET. Since 2003, he has been with BUET, where he is currently a Professor of Computer Science and Engineering Department. He was also a Visiting Associate Professor at the University of Fukui. His current research interests include evolutionary robotics, evolutionary computation, neural networks, machine learning, pattern recognition, and data mining. He has over 100 refereed publications in international journals and conferences.

Kazuyuki Murase received the M.E. degree in electrical engineering from Nagoya University, Nagoya, Japan, and the Ph.D. degree in biomedical engineering from Iowa State University, Ames, IA, USA, in 1978 and 1983, respectively.

From 1984, he was a Research Associate at the Department of Information Science, Toyohashi University of Technology, Toyohashi, Japan. He joined the Department of Information Science, Fukui University, Fukui, Japan, as an Associate Professor in 1988, and became a Professor in 1993. He served as the Chairman of the newly established Department of Human and Artificial Intelligence Systems of Fukui University in 1999. His current research interests include neuroscience of sensory systems, self-organizing neural networks, and bio-robotics.

Xin Yao (M'91–SM'96–F'03) is a Chair (Professor) of Computer Science and the Director of the Centre of Excellence for Research in Computational Intelligence and Applications, University of Birmingham, Birmingham, U.K. His current research interests include evolutionary computation and ensemble learning. He has over 400 refereed publications in international journals and conferences. He has been invited to give over 70 keynote/plenary speeches at international conferences.

Prof. Yao was the recipient of the 2001 IEEE Donald G. Fink Prize Paper Award, the 2010 IEEE Transactions on Evolutionary Computation Outstanding Paper Award, the 2010 BT Gordon Radley Award for Best Author of Innovation (Finalist), the 2011 IEEE Transactions on Neural Networks Outstanding Paper Award, and several other best paper awards at various conferences. He was also the recipient of the prestigious Royal Society Wolfson Research Merit Award in 2012 and was selected to receive the 2013 IEEE Computational Intelligence Society (CIS) Evolutionary Computation Pioneer Award. He was an Editor-in-Chief of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION from 2003 to 2008. He is a Distinguished Lecturer of the IEEE CIS.