Hindawi
Mathematical Problems in Engineering
Volume 2018, Article ID 2470171, 8 pages
https://doi.org/10.1155/2018/2470171

Research Article

Combining LSTM Network Ensemble via Adaptive Weighting for Improved Time Series Forecasting

Jae Young Choi[1] **and Bumshik Lee**[2]

[1]*Division of Computer & Electronic Systems Engineering, Hankuk University of Foreign Studies, Yongin-si, Republic of Korea*
[2]*Department of Information and Communications Engineering, Chosun University, Gwangju, Republic of Korea*

Correspondence should be addressed to Bumshik Lee; bslee@chosun.ac.kr

Received 6 April 2018; Revised 16 July 2018; Accepted 26 July 2018; Published 5 August 2018

Academic Editor: Wlodzimierz Ogryczak

Time series forecasting is essential for various engineering applications in finance, geology, and information technology, etc. Long Short-Term Memory (LSTM) networks are nowadays gaining renewed interest and they are replacing many practical implementations of the time series forecasting systems. This paper presents a novel LSTM ensemble forecasting algorithm that effectively combines multiple forecast (prediction) results from a set of individual LSTM networks. The main advantages of our LSTM ensemble method over other state-of-the-art ensemble techniques are summarized as follows: (1) we develop a novel way of dynamically adjusting the combining weights that are used for combining multiple LSTM models to produce the composite prediction output; for this, our method is devised for updating combining weights at each time step in an adaptive and recursive way by using both past prediction errors and forgetting weight factor; (2) our method is capable of well capturing nonlinear statistical properties in the time series, which considerably improves the forecasting accuracy; (3) our method is straightforward to implement and computationally efficient when it comes to runtime performance because it does not require the complex optimization in the process of finding combining weights. Comparative experiments demonstrate that our proposed LSTM ensemble method achieves state-of-the-art forecasting performance on four real-life time series datasets publicly available.

1. Introduction

Time series is a set of values wherein all values of one index are arranged in chronological order. The objective of *time series forecasting* is to estimate the next value of a sequence, given a number of previously observed values. To this end, forecast (prediction) models are needed to predict the future based on historical data [1]. The traditional mathematical (statistical) models, such as Least Square Regression (LSR) [2], Autoregressive Moving Average [3–5], and Neural Networks [6], were widely used and reported in literature for their utility in practical time series forecasting.

Time series forecasting has fundamental importance in numerous practical engineering fields such as energy, finance, geology, and information technology [7–12]. For instance, forecasting of electricity consumption is of great importance in deregulated electricity markets for all of the stakeholders: energy wholesalers, traders, retailers, and consumers [10]. The ability to accurately forecast the future electricity consumption will allow them to perform effective planning and efficient operations, leading to ultimate financial profits for them. Moreover, energy-related time series forecasting plays an important role in the planning and working of the power grid system [7, 8]; for instance, accurate and stable wind speed forecast has primary importance in the wind power industry and make an influence on power-system management and the stability of market economics [11].

However, the time series forecasting in the aforementioned applications is an inherently challenging problem due to the characteristics of dynamicity and nonstationarity [1, 6, 13, 14]. Additionally, any data volatility leads to increased forecasting instability. To overcome the above challenges, there is a growing consensus that *ensemble forecasting* [3–6, 13, 14], i.e., forecasting model combination, has advantage over using a single individual model in terms of enhancing forecasting accuracy. The most common approach of

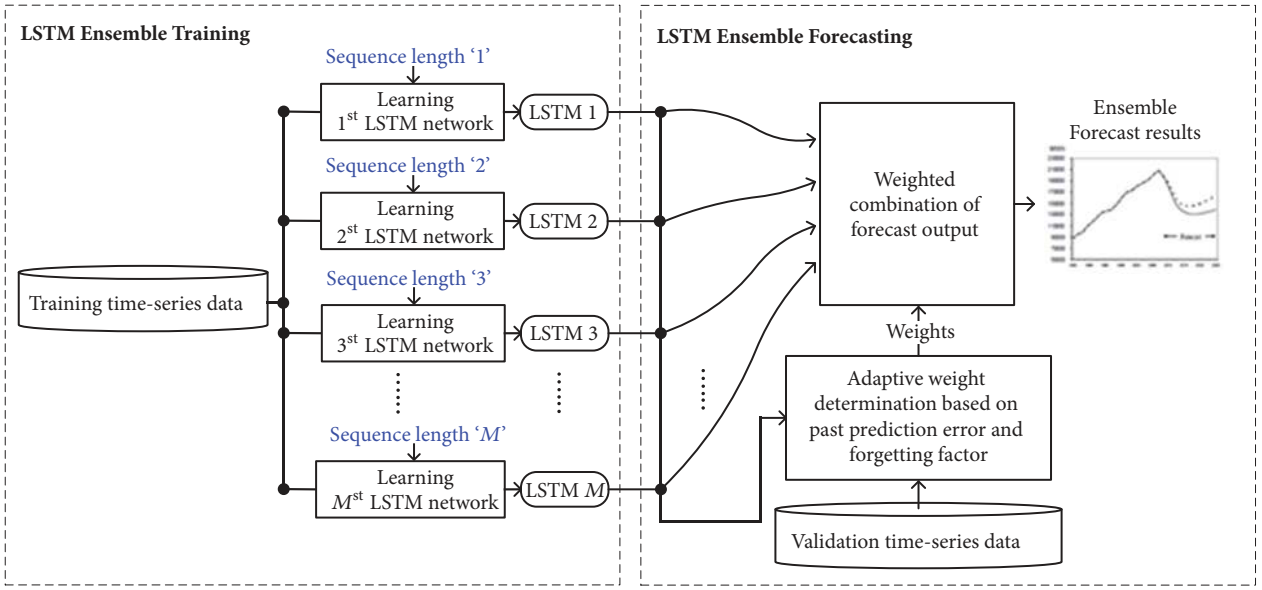

Figure 1: Overall framework of the proposed LSTM ensemble method for the purpose of time series forecasting.

ensemble forecasting is simple averaging that assigns equal weights to all forecasting component models [3–5]. The simple averaging approach is sensitive to extreme values (i.e., outliers) and unreliable for skewed distributions [6, 14]. To cope with this limitation, weighted combination schemes have been proposed. The authors in [2] proposed the Least Square Regression (LSR) that attempts to find the optimal weights by minimizing the Sum of Squared Error (SSE). The authors in [15] adopted the Average of In-sample Weights (AIW) scheme where each weight is simply computed as the normalized inverse absolute forecasting error of an individual model [16]. The authors in [6] developed a so-called Neural Network Based Linear Ensemble (NNLE) method that determines the combining weights through a neural network structure.

Recently, a class of mathematical models, called Recurrent Neural Networks (RNNs) [17], are nowadays gaining renew interest among researchers and they are replacing many kinds of practical implementation of the forecasting systems, previously based on statistical models [1]. In particular, Long Short-Term Memory (LSTM) networks—which are a variation of RNN—have proven to be one of the most powerful RNN models for time series forecasting and other related applications [1, 15]. The LSTM networks can be constructed in such a way that they are able to remember long-term relationships in the data. The LSTM networks have been shown to model temporal sequences and their long-range dependencies more accurately than original RNN model [1]. However, despite the recent popularity of the LSTM networks, their applicability in the context of ensemble forecasting has not been investigated yet. Hence, to our knowledge, how to best combine multiple forecast results of individual LSTM models still remains a challenging and open question.

In this paper, we present a novel *LSTM ensemble forecasting* method for improved time series forecasting, which effectively combines multiple forecasts (predictions)(throughout the remainder of this paper, both terms of "forecast" and "prediction" will be used interchangeably) inferred from the different and diverse LSTM models. Especially, we develop a novel way of dynamically adjusting the so-called combining weights that are used for combining multiple LSTM models to produce the composite prediction output. The main idea of our proposed method is to *update combining weights at each time step* in an adaptive and recursive way. For this, the weights can be determined by using both past prediction errors (measured up to the current time step) and forgetting weight factor. The weights are assigned to individual LSTM models, which improve the forecasting accuracy to a large extent. The overall framework of our proposed method is illustrated in Figure 1. Results show that the proposed LSTM ensemble achieves state-of-the-art forecasting performance on real-world time series dataset publicly available and it is considerably better than other recently developed ensemble forecasting methods as it will be shown in Section 4.

The rest of this paper is organized as follows. Section 2 describes our proposed approach for building an ensemble of LSTMs that is well-suited for use in time series forecasting. Then, we discuss how to find combining weights for the purpose of adaptively combining LSTM models. Section 4 presents and discusses our comparative experimental results. Finally, the conclusion is given in Section 5.

2. Building LSTM Ensemble for Time Series Forecasting

In the proposed method, an ensemble of LSTM networks should be first constructed in an effective way of maximizing a complementary effect during the combination of multiple LSTM forecast results, aiming to improve forecasting accuracy. Before explaining our LSTM ensemble construction,

we present a brief review on LSTM network for the sake of completeness as follows. LSTM networks and their variants have been successfully applied to time series forecasting [1]. LSTM networks are applied on sequential data as input, which without loss of generality means data samples that change over time. Input into LSTM networks involves a so-called *sequence length* parameter (i.e., the number of time steps) that is defined by the sample values over a finite time window [19]. Thus, sequence length is how we represent the change in the input vector over time; this is the time series aspect to the input data. The architecture of LSTM networks is generally composed of units called memory blocks. The memory block contains memory cells and three controlling gates, i.e., input gate, forget gate, and output gate. The memory cell is designed for preserving the knowledge of previous step with self-connections that can store (remember) the temporal state of the network while the gates control the update and flow direction of information.

For time series prediction of the input x_t, the LSTM updates the memory cell c_t and outputs a hidden state h_t according to the following calculations, which are performed at each time step t. The below equations give the full mechanism for a modern LSTM with forget gates [15]:

$$\begin{aligned} i_t &= \sigma\left(\mathrm{W}_{xi}x_t + \mathrm{W}_{hi}x_{t-1} + b_i\right), \\ f_t &= \sigma\left(\mathrm{W}_{xf}x_t + \mathrm{W}_{hf}x_{t-1} + b_f\right), \\ o_t &= \sigma\left(\mathrm{W}_{xo}x_t + \mathrm{W}_{ho}x_{t-1} + b_o\right), \\ c_t &= f_t \otimes c_{t-1} + i_t \otimes \phi\left(\mathrm{W}_{xc}x_t + \mathrm{W}_{hc}h_{t-1} + b_c\right), \\ h_t &= o_t \otimes \phi\left(c_t\right). \end{aligned} \tag{1}$$

In (1), W_{mn} denotes the weight of the connection from gate m and gate n, and b_n is bias parameter to learn, where $m \in \{x, h\}$ and $n \in \{i, f, o, c\}$. In addition, $\otimes$ represents the element-wise multiplication (Hadamard product), σ stands for the standard logistic sigmoid function, and ϕ denotes the *tanh* function: $\sigma(x) = 1/(1+e^{-x})$ and $\phi(x) = (e^{x} - e^{-x})/(e^{x} + e^{-x})$. The input, forget, and output gates are denoted by i_t, f_t, and o_t, respectively, while c_t represents the internal state of the memory cell c at time t. The value of the hidden layer of the LSTM at time t is the vector h_t, while h_{t-1} is the values output by each memory cell in the hidden layer at the previous time.

The underlying mechanism behind LSTM model (used for building our LSTM ensemble) mainly comes from the "gate" components (shown in (1)) that are designed for learning when to let prediction error in, and when to let it out. As such, LSTM gates provide an effective mechanism in terms of quickly modifying the memory content of the cells and the internal dynamics in a cooperative way [20, 21]. In this sense, the LSTM may have a superior ability to learn nonlinear statistical dependencies of real-world time series data in comparison to conventional forecasting models.

In the proposed ensemble forecasting method, each of the individual LSTM networks is used as "base (component) predictor model" [6]. Note that ensemble forecasting approach can be generally effective when there is considerable extent of diversities among individual base models, namely, ensemble members, [6, 13, 14]. In light of this fact, we vary the LSTM model parameter [15], namely, *sequence length*, to increase diversity among individual LSTM networks as ensemble members. For this, *we learn on each LSTM network for a particular sequence length*. Underlying idea for using different sequence length parameters is to inject *diversity* during the generation of individual LSTM models. This approach may be beneficial for effectively modelling highly nonlinear statistical dependencies, since using multiple sequence length values allows for creating multiple LSTM models with various number of memory cells, which is likely to provide complementary effect on learning the internal dynamics and characteristics of time series data to be predicted. In this way, an ensemble of LSTM models with *varying sequence length* is capable of *handling the dynamics and nonstationarities of real-world time series*.

Assuming that a total of M LSTM models in an ensemble are provided, their ensemble forecast result for time series, denoted as $(y^{(1)}, y^{(2)} \ldots, y^{(N)})$ with N observations, is given by

$$\hat{y}^{(k)} = \sum_{m=1}^{M} w_m \hat{y}_m^{(k)} \quad \text{for } k = 1, \ldots, N \tag{2}$$

where $\hat{y}_m^{(k)}$ denotes the forecast output (at the kth time step) obtained using the mth LSTM model and w_m is the associated combining weight. In (2), each weight w_m is assigned to a corresponding LSTM model's forecast output. Note that $0 \leq w_m \leq 1$ and $\sum_{m=1}^{M} w_m = 1$.

3. Finding Adaptive Weights for Combining LSTM Models

An important factor that affects the forecasting performance of our LSTM ensemble is a set of combining weights $w_m, m = 1, \ldots, M$. We develop a novel weight determination scheme which accounts for capturing the time varying dynamics of the underlying time series in an adaptive manner.

We now describe the details of our weight determination solution, which has been developed based on the following property: if the regression errors of individual estimators are uncorrelated and zero mean, their weighted averaging (shown in (2)) has minimum variance when the weights are inversely proportional to variances of the estimators [22]. This property provides the theoretical foundation for developing our weight determination solution; for details on this proof, please refer to the Appendix section. The combining weights are computed in the following recursive way:

$$w_m^{(k+1)} = w_m^{(k)} + \lambda \Delta w_m^{(k)} \quad \text{for } m = 1, \ldots, M \tag{3}$$

where we suggest $\lambda = 0.3$. Note that $\Delta w_m^{(k)}$ is computed based on the inverse prediction error of the respective LSTM base model as follows:

$$\Delta w_m^{(k)} = \frac{1/\varepsilon_m^{(k)}}{1/\varepsilon_m^{(1)} + 1/\varepsilon_m^{(2)} + \cdots + 1/\varepsilon_m^{(k)}} \tag{4}$$

Table 1: Description of the four different time series datasets [18] used in our experimentation.

Time Series	Type	Total size	Training size	Validation size	Testing size
River flow	Stationary, nonseasonal	600	360	90	150
Vehicles	Nonstationary, nonseasonal	252	151	38	63
Wine	Monthly seasonal	187	112	29	46
Airline	Monthly seasonal	144	86	22	36

The $\varepsilon_m^{(k)}$ is related to past prediction error measured up to the kth time step in the following way:

$$\varepsilon_m^{(k)} = \sum_{t=k-v+1}^{k} \gamma^{k-t} e_m^{(t)} \tag{5}$$

where $0 < \gamma \leq 1$, $1 \leq v \leq k$ and $e_m^{(t)}$ is the prediction error at each time step t of the mth LSTM model. In (5), $\varepsilon_m^{(k)}$ is calculated by defining a sliding time window formed by the last v prediction errors. Herein, the forgetting factor γ is devised for *reducing the influence of old prediction errors*. By performing weight update in time as described in (3), we can find the weights *by analyzing intrinsic patterns in successive data forecasting trials*; this would be beneficial for coping with nonstationary properties in the time series or avoiding complex optimization for finding adaptive weights.

Using (3), (4), and (5), the weights are updated over time series with T time steps ($k = 1, \ldots, T$), leading to the final weights $w_m^{(T)}$ ($m = 1, \ldots, M$). Finally, the weight to be assigned to each LSTM model in an ensemble is computed as follows:

$$w_m = \frac{w_m^{(T)}}{\sum_{n=1}^{M} w_n^{(T)}} \quad \text{for } m = 1, \ldots, M \tag{6}$$

Note that the weights computed using (6) satisfy the constraints such that $0 \leq w_m \leq 1$ and $\sum_{m=1}^{M} w_m = 1$.

4. Experimentation

4.1. Experimental Setup and Condition. The proposed LSTM ensemble method was implemented using TensorFlow [23] and was trained with the stochastic gradient descent (SGD). A total of ten LSTM base models (i.e., ensemble members) were used to build our LSTM ensemble. For each LSTM network, we used one hidden layer and the same number of memory cells as in the assigned sequence length value (e.g., LSTM with sequence length "5" has five memory cells). We set the γ and v parameters [shown in (5)] as 0.85 and 4, respectively. Also, the initial value of $w_m^{(k)}$ in (3) was set to zero.

We tested our proposed LSTM ensemble on four discrete time series datasets (please refer to Table 1), namely, "River flow", "Vehicles", "Wine", and "Airline" representing real-world phenomena which are publicly available at the Time Series Data Library (TSDL) [18]. For each time series, we used around 60% as training dataset, the successive 15% as validation dataset, and the remaining 25% as test dataset. Note that validation dataset was used for finding the combining weights and all the results reported here were obtained using test dataset.

Table 2: Demonstrating effectiveness of our proposed LSTM ensemble on improving forecasting performance. (a) MAE and (b) MSE.

(a)

Time series	Mean ± Std.	Best individual LSTM	Proposed LSTM ensemble
River Flow	0.67 ± 0.05	0.64	0.53
Vehicles	2.55 ± 0.32	2.22	1.85
Wine	1.27 ± 0.29	1.09	0.94
Airline	8.22 ± 1.81	6.43	5.58

(b)

Time series	Mean ± Std.	Best individual LSTM	Proposed LSTM ensemble
River Flow	1.31 ± 0.24	1.10	0.88
Vehicles	8.35 ± 1.97	6.25	4.45
Wine	9.63 ± 3.18	6.78	3.65
Airline	92.90 ± 15.83	81.73	64.11

Note: Mean±Std. indicates the average and standard deviation of MAE (or MSE) measures computed over all of the individual LSTM base models in an ensemble.

In our experimental study, we used the following two error measures used to evaluate the *forecasting accuracy*, namely, mean absolute error (MAE) and the mean square error (MSE) [1, 6], as follows:

$$\text{MAE} = \frac{\sum_{k=1}^{N} \left| y^{(k)} - \hat{y}^{(k)} \right|}{N} \tag{7}$$

$$\text{MSE} = \frac{\sum_{k=1}^{N} \left(y^{(k)} - \hat{y}^{(k)} \right)^2}{N} \tag{8}$$

where $y^{(k)}$ and $\hat{y}^{(k)}$ are the target and forecast values (outcomes), respectively, of time series with a total of N observations. Both MAE and MSE performances of the forecasting algorithms considered were computed based on one-step-ahead forecasts as suggested in [1, 6, 13].

4.2. Experimental Results

4.2.1. Effectiveness of the Proposed LSTM Ensemble Forecasting. The results in Table 2 demonstrate the effectiveness of our LSTM ensemble algorithm on improving forecasting

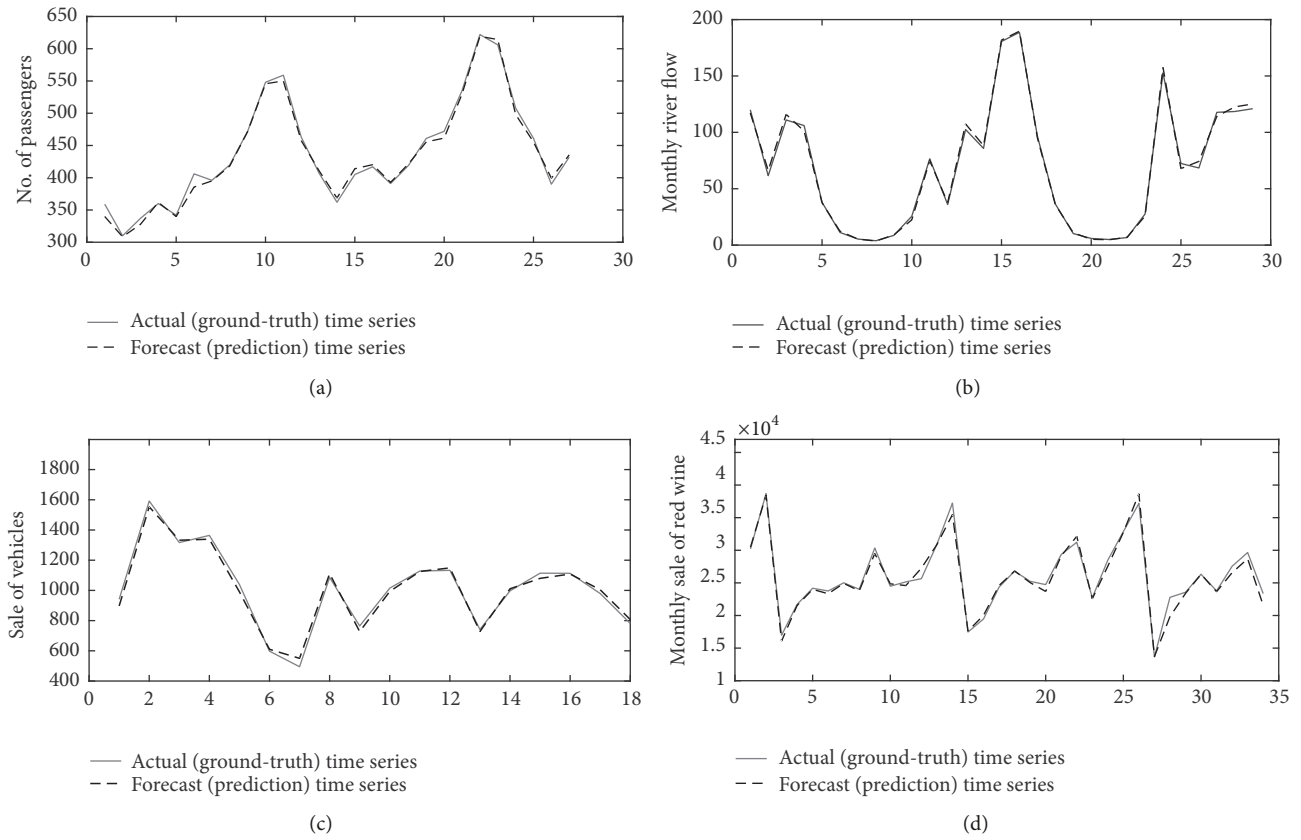

Figure 2: Time series forecasts of (a) Airline, (b) River flow, (c) Vehicles, and (d) Wine, obtained using our proposed LSTM ensemble forecasting algorithm.

performance (in terms of both MAE and MSE) against the case of using only a single individual LSTM base model. We can see that our LSTM ensemble greatly outperforms all the individual LSTM base models in an ensemble. To support this, compared to the best individual LSTM model, prediction error "MSE" can be significantly reduced using our ensemble method of up to about 22%, 29%, 46%, and 21% for respective "River flow", "Vehicles", "Wine", and "Airline" time series. Likewise, for using MAE, prediction errors can be reduced as much as 16%, 16%, 14%, and 13% in the same order of the aforementioned time series by using the proposed LSTM ensemble method.

Figure 2 depicts the actual and predicted time series obtained using our LSTM ensemble forecasting method. It is seen that, in each plot, the closeness between the actual observations and their forecasts is clearly evident. The results shown in Table 2 and Figure 2 confirm the advantage of our proposed LSTM ensemble method for notably improving forecasting accuracy compared to the approach of using a single individual LSTM network.

4.2.2. Comparison with Other State-of-the-Art Ensemble Forecasting Methods. We compared our proposed method with other state-of-the-art ensemble forecasting algorithms including simple averaging (Avg.) [3–5], Median [15], Least Square Regression (LSR) [2], Average of In-sample Weights (AIW) [16], and Neural Network Based Linear Ensemble (NNLE) [6]. Table 3 presents comparative results. Note that all the results for comparison were directly cited from corresponding papers recently published (for details, please refer to [6]). In Table 3, the proposed LSTM ensemble achieves the lowest prediction errors, namely, the highest prediction accuracy for both MAE and MSE measures. In particular, from Table 3, it is obvious that our LSTM ensemble approach can achieve the best performance for challenging time series "Airline" and "Vehicles", each of which is composed of nonstationary and quite irregular patterns (movements).

To further guarantee stable and robust comparison with other ensemble forecasting methods, we calculate the so-called their worth values [1, 6]. Note that the worth values are computed as the average percentage reductions in the forecasting errors of the worst one of all ensemble forecasting methods (under consideration) over all four time series datasets used. This measure shows the extent to which an ensemble forecasting method performs better than the worst ensemble and, hence, represents the overall "goodness" of the ensemble forecasting method. Let us denote by "$\text{error}_{i,j}$" the forecasting error (calculated via MAE or MSE) obtained for the jth ensemble method for the ith time series and by "max_error_i" the maximum (or worst) error obtained by a particular method for the ith time series at hand. Then, worth

Table 3: Comparison with other state-of-the-art ensemble forecasting methods. (a) MAE and (b) MSE.

(a)

Time series	Ensemble forecasting methods					
	Avg.[3–5]	Median[15]	LSR[2]	AIW[16]	NNLE[6]	Proposed
River Flow	0.751	0.676	0.736	0.749	0.638	**0.536**
Vehicles	2.087	2.139	2.059	2.071	2.001	**1.851**
Wine	2.075	2.173	2.466	2.372	1.923	**0.937**
Airline	11.63	11.73	10.68	10.22	7.434	**5.582**

(b)

Time series	Ensemble forecasting methods					
	Avg.[3–5]	Median[15]	LSR[2]	AIW[16]	NNLE[6]	Proposed
River Flow	1.158	1.138	1.245	1.156	0.978	**0.881**
Vehicles	6.188	6.683	7.508	6.175	5.531	**4.452**
Wine	9.233	10.05	10.07	9.204	7.524	**3.653**
Airline	181.4	176.5	158.3	143.1	86.63	**64.112**

Note: Bold values denote the best results of forecasting each time series.

Table 4: Comparison of the worth values obtained for different ensemble forecasting methods.

Methods	Worth values (%)	
	MAE	MSE
Avg. [3–5]	4.7848	8.2202
Median [15]	5.4671	5.6206
LSR [2]	3.6722	3.1835
AIW [16]	5.0325	13.6540
NNLE [6]	20.0354	31.3260
Proposed LSTM Ensemble	39.1271	49.5803

value, denoted as worth_j, for the jth ensemble forecasting method is defined as follows:

$$\text{worth}_j = \frac{1}{K}\sum_{i=1}^{K}\left[\left(\frac{\text{max\_error}_i - \text{error}_{i,j}}{\text{max\_error}_i}\right)\times 100\right] \quad \text{for } j = 1,\ldots P \tag{9}$$

where K and P are the total number of time series datasets and ensemble forecasting methods, respectively, used in our experiments. In (9), worth_j measures the percentage error reduction of the jth ensemble method, compared to the worst method. The worth values of all the ensemble methods for both MAE and MSE error measures are listed in Table 4. From this table, it can be seen that the proposed LSTM ensemble method achieves the best worth values, which are about 39% and 49% for MAE and MSE, respectively. This result indicates that, for four different time series, our LSTM ensemble forecasting method can provide around 39% and 49% more accurate results than a worst ensemble method in terms of MAE and MSE, respectively. The comparison results in Tables 3 and 4 validate the feasibility of our proposed LSTM ensemble method with regard to improving state-of-the-art forecasting performance.

4.2.3. Effect of LSTM Ensemble Size on Forecasting Accuracy. In the proposed method, the size of LSTM network ensemble (i.e., the number of LSTM models within an ensemble) is one of the important factors for determining the forecasting accuracy. In this subsection, we discuss experimental results by investigating the impact of our LSTM ensemble size on forecasting accuracy.

Figures 3(a) and 3(b) show the training and testing forecasting accuracy (in terms of MAE and MSE, respectively) with respect to different number of ensemble members for the "Wine" dataset. We can see that training forecasting accuracy for both MAE and MSE continues to increase as the size of LSTM ensemble becomes large and quickly levels off. Considering testing forecasting accuracy, it seems to generally improve as the size of ensemble increases up to a particular number and repeat increasing and decreasing, and finally converges to nearly the same constant value. It can be also observed that, in Figure 3, testing forecasting accuracy for both datasets is the best when the number of ensemble members is around 10. The above-mentioned observations indicate that larger LSTM ensemble size does not always guarantee improved (generalization [24]) forecasting accuracy. Moreover, the least number of ensemble members is more beneficial for reducing the computational costs required, especially when our proposed LSTM ensemble is applied to forecast (predict) a large number of time series, which is common in energy- and finance-related forecasting applications [7–9].

4.2.4. Runtime Performance. Furthermore, we assess the runtime performance of our LSTM ensemble forecasting framework using the "River flow" dataset. Our hardware configuration comprises a 3.3-GHz CPU and a 64G RAM with the NVidia Titan X GPU. It is found that the total time needed to train a single LSTM model is about 2.9 minutes, while the training time required to build our overall LSTM ensemble framework (for the case of ten ensemble members) is about 28.3 minutes. However, it

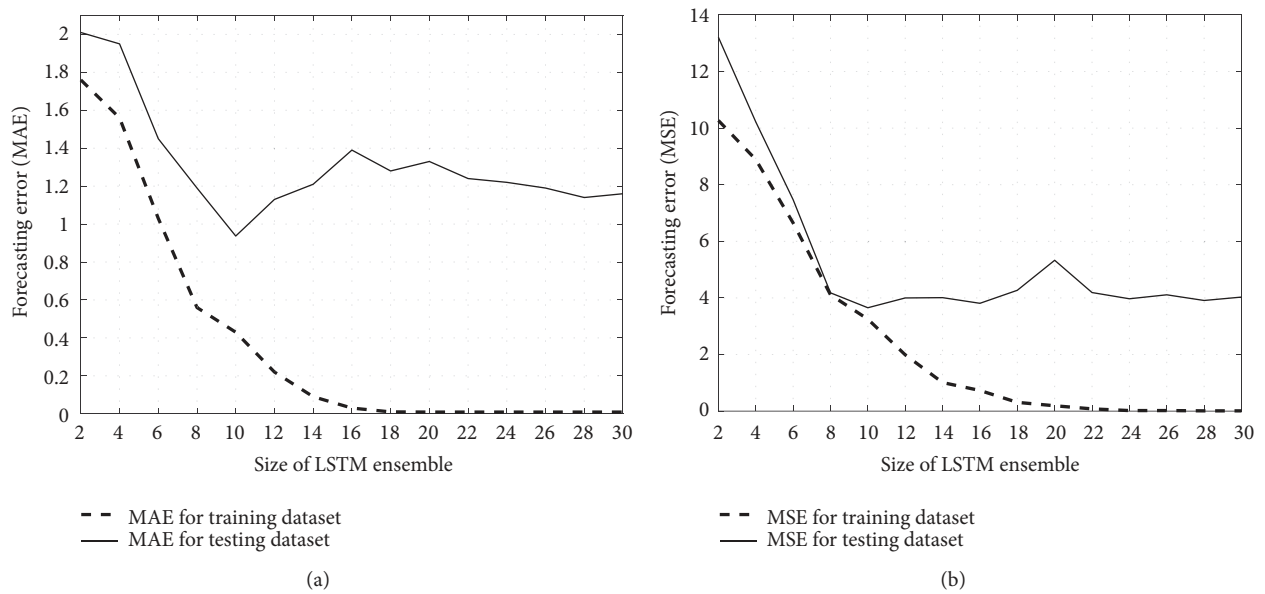

Figure 3: Impact of LSTM ensemble size in the proposed method on forecasting accuracy. (a) MAE and (b) MSE.

should be pointed out that the average time required to forecast across all time points (steps) is as low as 0.003 seconds. It should be also noted that the time required to construct our LSTM ensemble framework should not be considered in the measurement of the execution times because this process can be executed offline in real-life forecasting applications. In light of this fact, the proposed LSTM ensemble method can be feasible for practical engineering applications by considering the balance between forecasting accuracy, lower testing time, and straightforward implementation.

5. Conclusions

We have proposed a novel LSTM ensemble forecasting method. We have shown that our LSTM ensemble forecasting approach is capable of well capturing the dynamic behavior of real-world time series. Comparative experiments on the four challenging time series indicate that the proposed method achieves superior performance compared with other popular forecasting algorithms. This can be achieved by developing a novel scheme that can adjust combining weights based on time-dependent reasoning and self-adjustment. It is also shown that our LSTM ensemble forecasting can effectively model highly nonlinear statistical dependencies, since their gating mechanisms enable quickly modifying the memory content of the cells and the internal dynamics in a cooperative way [20, 21]. In addition, our complexity analysis demonstrates that our LSTM ensemble is able to have a runtime which is competitive with the approach to use only a single LSTM network. Consequently, our proposed LSTM ensemble forecasting solution can be readily applied in time series forecasting (prediction) problems, both in terms of forecasting accuracy and fast runtime performance.

Appendix

Weighted averaging defined in (2) obtains the combined output $\hat{y}^{(k)}$ by averaging the outputs $\hat{y}_m^{(k)}$, $m = 1, \ldots, M$, of individual estimator models with different weights. If we define prediction (or regression) error as $e_m^{(k)} = \hat{y}_m^{(k)} - y^{(k)}$ for each estimator, $\hat{y}^{(k)}$ can be expressed as

$$\hat{y}^{(k)} = \sum_{m=1}^{M} w_m \hat{y}_m^{(k)} = y^{(k)} + \sum_{m=1}^{M} w_m e_m^{(k)} \tag{A.1}$$

where the weights satisfy the constraint that $w_m \geq 0$ and $\sum_{m=1}^{M} w_m = 1$. The mean square error (MSE) of combined output $\hat{y}^{(k)}$ with respect to the target value $y^{(k)}$ can be written as follows [22]:

$$\mathrm{MSE}\left[\hat{y}^{(k)}\right] = \sum_{m=1}^{M} \sum_{n=1}^{M} w_m w_n C_{mn} \tag{A.2}$$

where C_{mn} stands for the symmetric covariance matrix defined by $C_{mn} = \mathrm{E}[e_m^{(k)} e_n^{(k)}]$. Our goal is to find the optimal weights that minimize the aforementioned MSE, which can be solved by applying the *Lagrange* multiplier as follows [22]:

$$\frac{\partial}{\partial w}\left[\sum_{n=1}^{M} w_m w_n C_{mn} - 2\lambda\left(\sum_{m=1}^{M} w_m - 1\right)\right] = 0. \tag{A.3}$$

By imposing the constraint $\sum_{m=1}^{M} w_m = 1$, we find that

$$w_m = \frac{\sum_{n=1}^{M} C_{mn}^{-1}}{\sum_{k=1}^{M} \sum_{n=1}^{M} C_{kn}^{-1}}. \tag{A.4}$$

Under the assumption that the errors $e_m^{(k)}$ are uncorrelated and zero mean, i.e., $C_{mn} = 0 \; \forall \; m \neq n$, and the optimal w_m can be obtained [22]

$$w_m = \frac{\sigma_m^{-2}}{\sum_{n=1}^{M} \sigma_n^{-2}} \tag{A.5}$$

where $\sigma_m^2 = C_{mm} = \mathrm{E}[(e_m^{(k)})^2]$. The result in (A.5) shows that the weights should be inversely proportional to variances (i.e., errors) of the respective estimators. In other words, it means that the weights should be in proportion to the prediction (regression) performance of individual estimators.

Data Availability

The datasets generated and/or analyzed during the current study are available from the corresponding author on reasonable request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT [NRF-2017R1C1B1008646]. This work was supported by Hankuk University of Foreign Studies Research Fund. This work was also supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (no. 2015R1D1A1A01057420).

References

[1] F. M. Bianchi, E. Maiorino, M. C. Kampffmeyer, A. Rizzi, and R. Jenssen, "An overview and comparative analysis of recurrent neural networks for short term load forecasting," 2017.

[2] R. Adhikari and R. K. Agrawal, "Performance evaluation of weights selection schemes for linear combination of multiple forecasts," *Artificial Intelligence Review*, vol. 42, no. 4, pp. 529–548, 2012.

[3] V. R. R. Jose and R. L. Winkler, "Simple robust averages of forecasts: Some empirical results," *International Journal of Forecasting*, vol. 24, no. 1, pp. 163–169, 2008.

[4] J. G. de Gooijer and R. J. Hyndman, "25 years of time series forecasting," *International Journal of Forecasting*, vol. 22, no. 3, pp. 443–473, 2006.

[5] S. Makridakis and R. L. Winkler, "Averages of forecasts: some empirical results," *Management Science*, vol. 29, no. 9, pp. 987–996, 1983.

[6] R. Adhikari, "A neural network based linear ensemble framework for time series forecasting," *Neurocomputing*, vol. 157, pp. 231–242, 2015.

[7] R. K. Jain, K. M. Smith, P. J. Culligan, and J. E. Taylor, "Forecasting energy consumption of multi-family residential buildings using support vector regression: Investigating the impact of temporal and spatial monitoring granularity on performance accuracy," *Applied Energy*, vol. 123, pp. 168–178, 2014.

[8] V. P. Utgikar and J. P. Scott, "Energy forecasting: Predictions, reality and analysis of causes of error," *Energy Policy*, vol. 34, no. 17, pp. 3087–3092, 2006.

[9] W. Huang, K. K. Lai, Y. Nakamori, S. Wang, and L. Yu, "Neural networks in finance and economics forecasting," *International Journal of Information Technology & Decision Making*, vol. 6, no. 1, pp. 113–140, 2007.

[10] F. Kaytez, M. C. Taplamacioglu, E. Cam, and F. Hardalac, "Forecasting electricity consumption: a comparison of regression analysis, neural networks and least squares support vector machines," *International Journal of Electrical Power & Energy Systems*, vol. 67, pp. 431–438, 2015.

[11] H. Yang, Z. Jiang, and H. Lu, "A Hybrid Wind Speed Forecasting System Based on a 'Decomposition and Ensemble' Strategy and Fuzzy Time Series," *Energies*, vol. 10, no. 9, p. 1422, 2017.

[12] J. Kitchen and R. Monaco, "Real-time forecasting in practice: The U.S. Treasury Staff's Real-Time GDP Forecast System," *Business Economics: The Journal of the National Association of Business Economists*, vol. 38, no. 4, pp. 10–19, October 2003.

[13] N. Kourentzes, D. K. Barrow, and S. F. Crone, "Neural network ensemble operators for time series forecasting," *Expert Systems with Applications*, vol. 41, no. 9, pp. 4235–4244, 2014.

[14] P. S. Freitas and A. J. Rodrigues, "Model combination in neural-based forecasting," *European Journal of Operational Research*, vol. 173, no. 3, pp. 801–814, 2006.

[15] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[16] J. S. Armstrong, Ed., *Principles of forecasting: a handbook for researchers and practitioners*, vol. 30, Springer Science & Business Media, New York, NY, USA, 2001.

[17] T. G. Barbounis, J. B. Theocharis, M. C. Alexiadis, and P. S. Dokopoulos, "Long-term wind speed and power forecasting using local recurrent neural network models," *IEEE Transactions on Energy Conversion*, vol. 21, no. 1, pp. 273–284, 2006.

[18] "Time Series Data Library (TSDL)," http://robjhyndman.com/TSDL/.

[19] Z. C. Lipton, J. Berkowitz, and C. Elkan, "A critical review of recurrent neural networks for sequence learning," https://arxiv.org/abs/1506.00019.

[20] V. Veeriah, N. Zhuang, and G.-J. Qi, "Differential recurrent neural networks for action recognition," in *Proceedings of the 15th IEEE International Conference on Computer Vision, ICCV 2015*, pp. 4041–4049, Chile, December 2015.

[21] N. Srivastava, E. Mansimov, and R. Salakhutdinov, "Unsupervised learning of video representations using LSTMs," in *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015*, pp. 843–852, France, July 2015.

[22] M. P. Perrone and L. N. Cooper, "When networks disagree: Ensemble methods for hybrid neural networks," in *How We Learn; How We Remember: Toward an Understanding of Brain and Neural Systems: Selected Papers of Leon*, vol. 10, pp. 342–358, World Scientific, 1995.

[23] M. Abadi, A. Agarwal, P. Barham et al., *Tensorflow: Large-scale machine learning on heterogeneous distributed systems*, 2016.

[24] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, New York, NY, USA, 2006.