IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON RECENT ADVANCES ON HYBRID COMPLEX NETWORKS: ANALYSIS AND CONTROL

Received October 13, 2020, accepted November 6, 2020, date of publication November 16, 2020, date of current version November 25, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3037681

Forecasting Stock Price Based on Frequency Components by EMD and Neural Networks

WANGWEI SHU, (Graduate Student Member, IEEE), AND QIANG GAO, (Senior Member, IEEE)
Department of Electronic and Information Engineering, Beihang University, Beijing 100191, China

Corresponding author: Wangwei Shu (shuwangwei@buaa.edu.cn)

This work was supported by the School of Electronics and Information Engineering, Beihang University.

ABSTRACT Predicting stock price based on the features of raw data has been a significant but challenging task for researchers. Various frequency components of the raw stock price series represent characteristics of stock prices in different time scales. Therefore, it makes sense for predicting stock prices to take these frequency components into account. In this paper, a novel hybrid model is proposed to predict stock prices, which combines empirical mode decomposition (EMD), convolutional neural network (CNN) and Long Short-Term Memory (LSTM). For this purpose, the original stock price series are first decomposed into a finite number of intrinsic mode functions (IMFs) under different frequencies by EMD. For each component, a CNN is used to extract the features. Then through a LSTM network, the temporal dependencies of all extracted features are modeled and the final predicted prices are obtained after a linear transformation. Two prediction steps, one day and one week, of Shanghai Stock Exchange Composite Index (SSE) are used to test the proposed model. The experimental results show that the hybrid network can achieve better performances by modeling different frequencies compared with other state-of-the-art models.

INDEX TERMS EMD, CNN, LSTM, multi-frequency modeling.

I. INTRODUCTION

Stock price prediction is an important issue to stock investors to seek profit-maximization strategies [1]–[4]. Since the stock price series are non-linear and non-stationary, it is a challenging task to predict the stock prices reliably and accurately [5]. Recently, with the development of machine learning, artificial neural networks (ANN) has become an effective tool for analyzing and predicting time series [6], [7] because of its nonlinear modeling capability.

Traditional ANN lacks the ability to model the long-term dependency of time series, which promotes the proposing of Long Short-Term Memory (LSTM) network, a gated memory cell. Since the predicted stock price is not only related to the stock price at the current time, but also to the data at earlier time. LSTM obtains time dependencies of the before-after associated data through its recurrent structure, which makes LSTM to be an effective tool to predict time series. Furthermore, LSTM filters the information selectively through a ''gate'' structure to extract more useful information from historical data in training [1].

Investors trading at different frequencies result in different levels of stock price volatility [13]. For those looking for long-term gains, it relies more on low-frequency information in stock price series to predict future stock prices, and vice versa. Therefore, it is significant to reveal the multi-frequency characteristics of the stock price time-series. Both discrete Fourier transform (DFT) and empirical mode decomposition (EMD) are effective tools to decompose time series and they have been applied in previous researches. However, DFT requires the time series to be stationary and linear, which are not met in the stock price series.

Empirical mode decomposition (EMD), proposed by Huang *et al.* [14], is a method for proceeding nonlinear and none-stationary time series, which decomposed original data into a finite number of intrinsic mode functions (IMFs) and a residue, which contain information about different frequencies. Some researchers transform the complicated series into a finite number of simple series through EMD, which are more likely to be modeled by neural network. But they just model the sub-series as new series and neglect the information at different frequencies. Meanwhile, the proceedings of the researches are based on the assumption that the decomposed values at the current time are independent of future information. Therefore, they decompose full series at once and don't take the effect of future prices on the decomposed components into account.

The associate editor coordinating the review of this manuscript and approving it for publication was Jianquan Lu.

As one of the most successful deep learning methods, convolutional neural network (CNN) can extracts features of the input data through convolutional kernels [27]. By kernels with different lengths, CNNs can obtain features under different frequencies after the original series are decomposed into frequency components.

In this paper, we divide the original stock price series instance into four sub sequences based on the temporal relationship and then each sequence is decomposed through EMD. In order to obtain the frequency representations, a CNN is used for each component. Because of the capability of temporal modeling, LSTM learns the temporal dependencies of features extracted by all CNNs. We get the predicted price after a linear transform of the output of the LSTM. The original data in this paper comes from samples of Shanghai Stock Exchange Composite Index (SSE) data.

II. RELATED WORK

There were researches using ANN to deal with time series. Adebiyi *et al.* [8] used ANN to predict public stock data trends and compared the performance with autoregressive integrated moving average (ARIMA). Wang *et al.* [9] combined particle swarm optimization (PSO) and BP neural network to predict gold price. Kuremoto *et al.* [10] forecasted time series using a deep belief network with restricted Boltzmann machines. Because of the capacity of temporal modeling, Xiong *et al.* [11] used LSTM with 27 features to predict the S&P500 volatility. Saxe *et al.* [12] proposed a novel LSTM-AR hybrid model.

CNN was also widely applied in researches modeling time series. Sainath *et al.* [29] used CNN to extract features under different frequencies of English-spoken signals and then used LSTM to classify the spoken words. Li *et al.* [27] combined 1D-CNN and LSTM to predict PM2.5 index. In financial time series prediction, Vidal and Kristjanpoller [28] predicted gold volatility through modeling Markov Transition Field (MTF) by multilayer CNNs.

In order to use frequency characteristics, Zhang and Li [13] decomposed the memory states of LSTM to a set of K discrete frequencies inspired by DFT. However, financial series are non-linear and non-stationary, but DFT requires the time series to be stationary and linear [14]. Furthermore, the research set the number of discrete frequencies unchanged regardless of the length of series, which might cause time domain aliasing. Yu *et al.* [15] decomposed the crude oil price series into nine IMFs and one residue, and predicted each component by a forward neural network. Cao and Li [1] decomposed the stock price series and predicted each IMF by a multi-layer LSTM network.

III. METHODOLOGY

This section details the EMD algorithm and two neural networks including CNN and LSTM, and then describes the architecture of the proposed EMD-CNN-LSTM hybrid model.

A. EMD

EMD decomposed the original data into a collection of IMFs and a residue based on the local characters of the time series, including local maxima, local minima, and zero-crossings [16]. There are two conditions that an IMF satisfies: (1) the number of extrema and the number of zero crossings must be differ at most by one; and (2) the mean value of the envelope defined by the local maxima and the envelope defined by the local minima must be zero at any point. The method for decomposing is called sifting process [14]

(1) Given a TS $x(t)$, identify all the maxima and minima, and form all maxima to an upper envelope $u(t)$ and all minima to a lower envelope $l(t)$ through interpolation.

(2) Calculate the mean value of upper and lower envelopes $m(t) = (u(t) + l(t))/2$.

(3) Subtract $m(t)$ from $x(t)$ to get a detailed component $h(t) = x(t) - m(t)$.

(4) Repeat step (1) to step (3) with $h(t)$ as a new input until the $h(t)$ satisfies the two conditions mentioned above or the number of iterations reaches the user defined maximum iteration, and $h(t)$ is defined as $c_1(t)$ as the first IMF.

(5) Separate $c_1(t)$ from $x(t)$ and get a new sequence without high frequency components $r_1(t) = x(t) - c_1(t)$.

(6) Repeat step (1) to (5) for $r_1(t)$ until all IMFs and residue are obtained.

The original TS is decomposed as:

$$x(t) = \sum c_i(t) + r(t) \tag{1}$$

Rilling *et al.* [17] proposed two threshold-based stopping criteria

$$\frac{\text{count}(t|\delta(t) < \theta_1)}{c(t)} \geq 1 - \alpha \tag{2}$$

$$\delta(t) < \theta_2 \tag{3}$$

$$\delta(t) = |\frac{u(t) + l(t)}{u(t) - l(t)}| \tag{4}$$

We typically set $\alpha = 0.05$, $\theta_1 = 0.05$ and $\theta_2 = 0.05$.

B. CNN AND LSTM

1) CNN

A CNN uses generally raw data as input instead of handcrafted features. The input data is processed through trainable convolutional layers for learning an appropriate representation of the input [18], [25]. In this paper, the input to an CNN is a frequency component. The network is designed to learn a collection of parameters to represent the frequency information of the component.

For the convolutional layer, the operation of $i-th$ layer can be expressed as:

$$T_i = act(W_i \otimes C_i + b_i) \tag{5}$$

where $\otimes$ denotes convolution operation, W denotes a set of N one-dimensional kernels which are used for extract the frequency features of the components, b is a bias vector, and act is the activation function. C_i denotes the $i-th$ component.

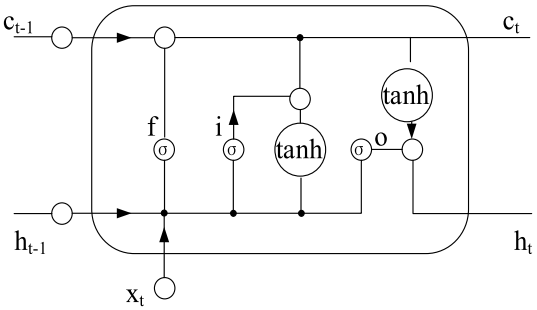


FIGURE 1. LSTM unit structure.

2) LSTM

Long Short-Term Memory is a variant of Recurrent Neural Network (RNN) proposed by Hochreiter and Schmidhuber [19], which is capable to handle long-term dependency in a time series. LSTM unit structure is shown in Figure 1.

At time t, x_t is the input vector, c_t is the memory state vector and h_t is the output vector from c_t. Then LSTM can be formulated as follow:

$$i_t = sigmoid(W_i x_t + U_i h_{t-1} + b_i) \quad (6)$$
$$f_t = sigmoid(W_f x_t + U_f h_{t-1} + b_f) \quad (7)$$
$$\tilde{c}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c) \quad (8)$$
$$c_t = i_t \circ \tilde{c}_t + f_t \circ c_{t-1} \quad (9)$$
$$o_t = sigmoid(W_o x_t + U_o h_{t-1} + b_o) \quad (10)$$
$$h_t = o_t \circ \tanh(c_t) \quad (11)$$

where i_t is the input gate, f_t is the forget gate, and o_t is the output gate. W_* and W_* are trainable weight matrices, b_* are trainable bias vectors, and $\circ$ denotes matrix multiplication by elements.

Because of its hidden states evolving themselves over time and its three control gates, LSTM can obtain long-term dependencies of the input series easily. In this paper, we set the features extracted by CNN as the input of LSTM to model frequency features in chronological order. Instead of capturing the time dependencies of raw data, we model the frequency features – time features through LSTM.

C. PROPOSED MODEL

In Figure 2, each instance contains 1000 10-minute sample values of SSE, which is denoted as $[x_1, x_2, \ldots, x_{1000}]$. We divide every instance to 4 sub-series by every 250 points. Four sub-series are input into the model in chronological order.

First, we decompose the first sub-series into several IMFs and one residue through EMD and five frequency components are obtained after all components are aligned. Specifically, for each component, we use a one-dimensional convolutional layer with 4 feature maps. The neurons in a convolutional layer are connected only to a small region of the previous layer called a receptive field. The size of the receptive field are determined by the length of kernels. For all convolutional layers, we use kernels of different

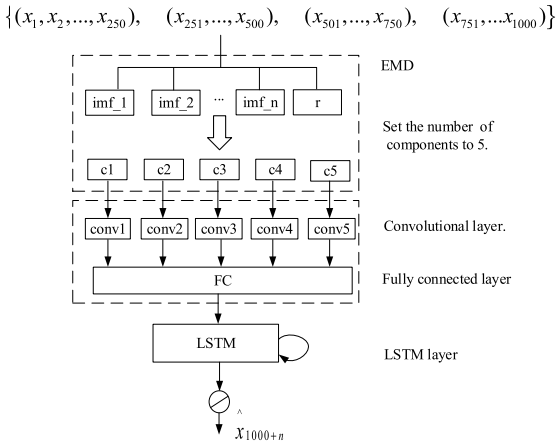


FIGURE 2. Structure of proposed model.

lengths to extract features at different time scales of frequency components. There is edge effect in the frequency components when the boundary of decomposed subseries achieves extreme value. We predict the future prices based on the features of frequency components extracted by CNN instead of frequency components themselves. Therefore, the edge effect of extreme value in the boundary of the decomposed subseries can be significantly reduced.

The stride of convolutional operation should be determined. If the convolutional stride is too small relative to the length of the kernel, there will be a lot of repeated calculations in the convolution calculation and redundant output information. Meanwhile, if the stride is too large, the feature fluctuations will be relatively large, which will result in the learning difficulty in the next stage of our model. Experiments with different convolutional strides have been conducted and the results show that a quarter of the kernel length is optimal.

After the convolutional layer, we use a fully-connected (FC) layer with 256 neurons to combine all features. The outputs of FC can be regarded as features in frequency domain of the first sub-series, which are used as an input vector of LSTM network at time $t = 0$. The number of hidden units of LSTM is set to 32. Then the 3 sub-series rest are input to the model one by one so that LSTM layers gets the temporal dependencies of frequency features extracted from four sub-series. After a linear transform of the final outputs of LSTM, we get the predicted stock price finally.

The activation function of convolutional layer and FC layer are "Relu" function and "tanh" function. We use mean absolute error (MAE) as the loss function.

$$Loss = MAE = \frac{1}{M}\sum_{m=1}^{M} |\hat{y}_m - y_m| \quad (12)$$

where $\hat{y}_m$ denotes predicted value, and y_m denotes the original value corresponding to the m-th instance. To speed up the training process, we use Adam optimizer algorithm [20] and the learning rate is set to 1e-3.

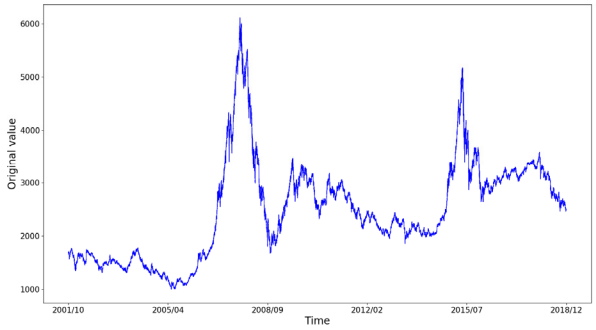


FIGURE 3. 10-minute samples of SSE.

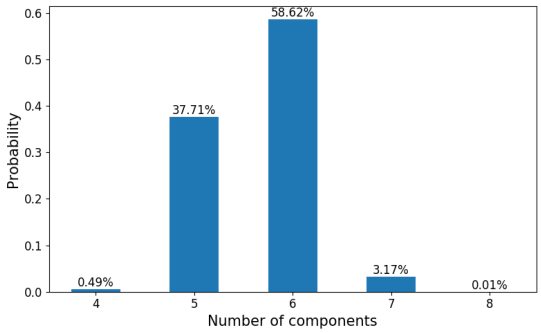


FIGURE 4. Frequency distribution histogram of component numbers under Rilling stop criterions.

10-minute samples from October 29, 2001 to December 28, 2018 of SSE are used as our dataset. Figure 3 shows the original SSE data, which is volatile and non-stationary in short term. Every instance consists of 1000 consecutive points, and the $(1000 + n) - th$ point is set as label of the instance. 80% of instances are used for training, 10% of instances are used for validating and 10% of instances are used for testing after all of instances are disrupted.

IV. EXPERIMENTS

In this section, we conduct experiments to test our model. First, we set the number of components to 5 according to the numbers of IMFs of all instances, and then we discuss the impact of single frequency component on prediction performance. Finally, we present the comparison between proposed model and other methods. Two steps of prediction, 24-step (one day) and 120-step (one week), are conducted in this section.

Time series of SSE with a frequency of 10 minutes is used in our experiments. Since prices one day ahead are predicted, the time granularity of historical information smaller than one day is preferred. In addition, we divide historical price series of two months into sub-series of two weeks and then decompose the sub-series into frequency components. There should be enough data points in sub-series to make proper frequency decomposition.

The experiments are conducted on a Windows 10 PC with Intel Core i5 CPU and the program is run on python 3.5. The proposed model is built and run on TensorFlow 1.12.0, a machine learning platform.

A. THE PROCESSING OF DATA

To ensure the parameters of our network can be updated effectively and speed up the training process [22], we normalize original data into $[-1, 1]$. The formula for normalization is as follow:

$$x(t)' = \frac{x(t) - mean(x(t))}{\max |x(t) - mean(x(t))|} \tag{13}$$

where $x(t)$ denotes raw data and $x(t)'$ denotes normalized data. We can restore the predicted price by the formula (14).

$$\hat{x}(t) = \max |x(t) - mean(x(t))| * \hat{x}(t)' + mean(x(t)) \tag{14}$$

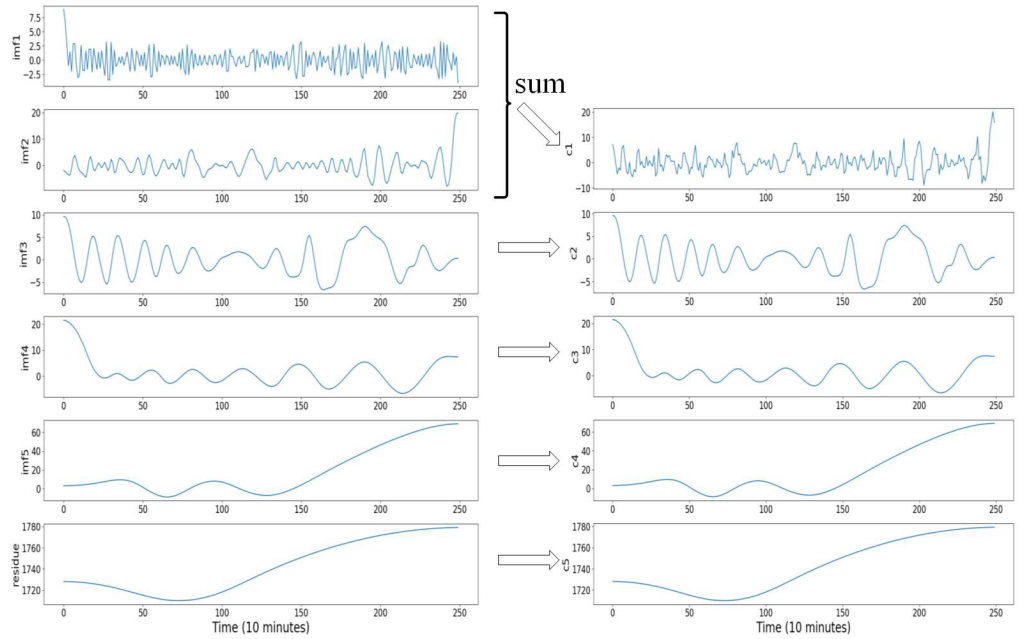
EMD is an adaptive algorithm based on local characters and the processing of decomposing will not stop until stop criterions are reached, which means the number of components is unsure before the process is finished. For our model, however, the number of convolutional networks is set unchanged. To make the number of components compatible with the model, we decompose all sub-series of all samples and under stop criterions mentioned at section II and count components of all sub-series. The probability distribution histogram is shown in figure 4. More than 96% of sub-series are decomposed to five or six components at Rilling stop criterions. In order to obtain sufficient frequency components and reduce the model parameters, we set the number of frequency components to 5 instead of 6.

Considering the large amount of noise in financial time series, which is mainly contained in high-frequency components [21], we align the low frequency components of all sub-series and combine all high frequency components into one component. For example, as is shown in figure 5 (a), sub-series s_1 has six components under default stop criterions, and we combine IMF1 and IMF2 as the first component, with the remaining four unchanged. For those with 4 components at default stop criterions, we set the 4-th component to a zero-setting sequence as the figure 5 (b) shows. Finally, all sub-series are decomposed to five components. Because the values can be very close to zero, we normalize the five components to $[-1, 1]$ by the formula (13) respectively.

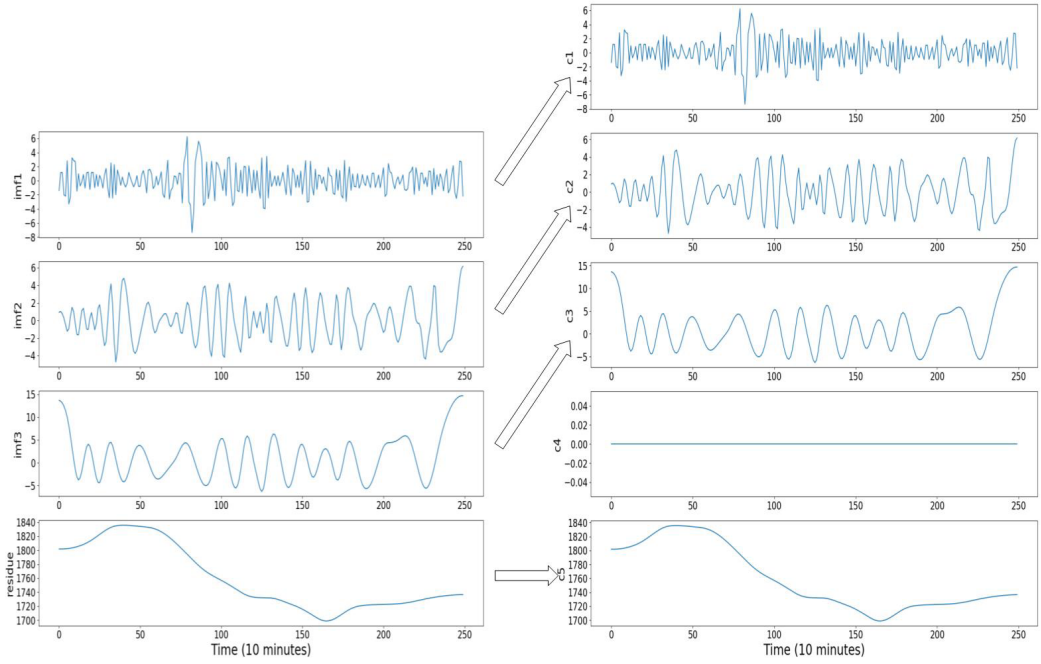
B. RESULTS OF PROPOSED MODEL

We extract the frequency features through several kernels in convolutional layer. The length of kernels determines the time scale on which frequency features are extracted. To learn the impact of time scale on predictive performance, we first use convolutional kernels of the same length for the five components to predict prices one day and one week later, and the best hyperparameters of length are obtained. Then we adjust the length of five components individually until we get the best predictive performance. The results are shown in Table 1.

Prediction steps are shown in the first column in Table 1, lengths of kernels from the 1-st to the 5-th component, the order of which are from higher to lower frequency



(a) Combine high frequency components



(b) Set the 4-th component to zeros

FIGURE 5. Alignment of frequency components.

component, are shown in the second column, and the MAE is shown in the third column. And root mean square error (RMSE) is shown in the fourth column. In both prediction tasks, kernels of length 64 perform best when all kernels' length are consistent. In addition, MAE under kernels with different lengths is lower than the other one, and the best hyperparameters in this condition follow the order from smallest to biggest. For high frequency components, smaller kernels lead to a better performance, and vice versa. The residue, the lowest frequency component, represents the tendency information of the series in the longest period as figure 5 shows. Therefore, it needs a bigger receptive field to extract the features. Components representing high frequency information have smaller periods, so smaller kernels extracting periodic fluctuation information at higher frequencies are required. What's more, for the same component, the length of kernels to predict prices after one day are equal to or smaller than that to predict prices after one week. For a short-term prediction, features on a smaller time scale are more important. For example, tendency and volatility features

TABLE 1. Prediction mean absolute error of different lengths of convolution kernels.

Prediction steps	Lengths of convolutional kernels	MAE	RMSE
1-day	16,16,16,16,16	21.88	40.11
	32,32,32,32,32	20.94	39.61
	64,64,64,64,64	19.16	39.11
	96,96,96,96,96	19.67	39.72
	64,64,64,64,128	18.95	37.13
	64,64,64,64,48	19.74	39.28
	64,64,64,48,128	18.84	37.66
	64,64,64,80,128	20.58	39.71
	64,64,48,64,128	18.44	37.33
	64,64,80,64,128	19.38	39.55
	64,48,48,64,128	18.11	36.02
	64,80,48,64,128	18.66	36.76
	32,48,48,64,128	18.37	35.99
	16,48,48,64,128	**17.61**	**35.04**
	80,48,48,64,128	19.15	39.48
1-week	16,16,16,16,16	26.88	45.76
	32,32,32,32,32	23.76	43.63
	64,64,64,64,64	22.82	42.04
	128,128,128,128	33.87	58.36
	64,64,64,64,32	24.92	43.31
	64,64,64,64,96	23.87	41.55
	64,64,64,64,128	21.73	39.99
	64,64,64,32,128	22.82	42.11
	64,64,64,96,128	24.16	43.27
	64,64,32,64,128	30.57	55.49
	64,64,96,64,128	23.43	41.54
	64,32,64,64,128	26.73	45.38
	64,96,64,64,128	25.54	43.86
	32,64,64,64,128	21.56	40.17
	16,64,64,64,128	**21.33**	**39.88**
	80,64,64,64,128	22.65	40.92

in recent months are less important than that in recent days to predict stock prices on next day.

C. IMPACT OF SINGLE FREQUENCY COMPONENT

To study the relationship between different components and prediction performance, we conduct experiment by removing the highest and the lowest frequency component separately, and the results are shown in Table 2. For both predictions, removing the lowest frequency component makes the prediction performance worse significantly. For one-day step prediction, removing the highest frequency component makes MAE larger by 12.3%, and for one-week step prediction, it makes MAE smaller by 6.9%.

The experimental results show the following information. Firstly, the lowest frequency components of a stock prices series, which represents the tendency of the series, contains important sequence feature information. When we remove these components, we lost amount of valid information of

TABLE 2. Comparison with removing two components separately.

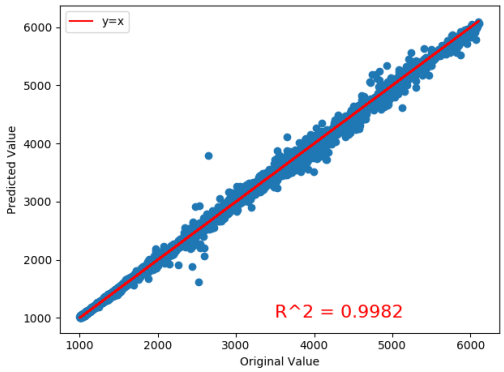
Prediction steps	The component removed	MAE	RMSE
1-day	**None**	**17.61**	**35.04**
	Highest frequency component	19.78	37.84
	lowest frequency component	157.59	417.32
1-week	None	21.33	39.88
	Highest frequency component	**19.88**	**39.60**
	lowest frequency component	160.09	453.21

the series. Secondly, because of the high volatility, the stock price series contain a lot of noise, which is mainly represented in high-frequency components. However, high-frequency components contain not only noise, but also a number of features at short time scales, which are of great importance to make short-term forecasts. However, high-frequency components contain not only noise, but also a number of features at short time scales, which are of great importance to make short-term forecasts. Therefore, removing the highest frequency component makes the performance worse in one-day step prediction. Finally, for long-term prediction, features at short time scales can be irrelevant. Removing the highest frequency component reduces the effects of irrelevant features and noise, and the prediction performance becomes better then. For investors with long trading cycles, they do not pay attention to short-term price fluctuation characteristics, such as price changes caused by high-frequency traders, and the fluctuation can even mislead these investors. On the contrary, for high-frequency traders, they care not only short-term price fluctuation characteristics, but also the price trend at a longer period to make their single stock transaction more profitable. Therefore, when we make long-term price prediction or stock transaction, it is better to filter out high-frequency noise firstly.

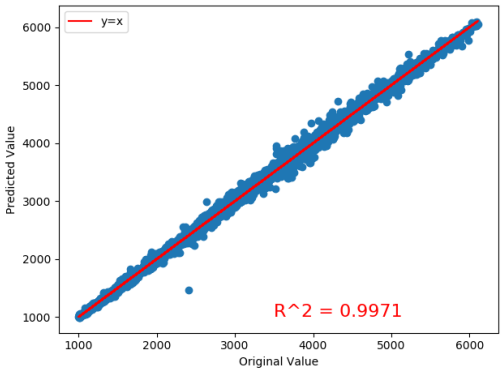
D. COMPARISON WITH OTHER MODELS

In this section, to evaluate the prediction performance of proposed model, we compared several prediction models, including persistence, LSTM, support vector regression (SVR), EMD-LSTM, CNN-LSTM and EMD-SVR. In all models, the same dataset was used.

Persistence model regard the price at current time as the predicted price. LSTM network predicts the original series directly through extracting the time dependencies of raw data. In EMD-LSTM network, we decompose the original data into 5 components in the same way as the proposed model. Then we predict each component through a LSTM. Five predicted values are combined as the final predicted price after a linear



(a) Linear regression analysis of 1-day prediction



(b) Linear regression analysis of 1-week prediction

FIGURE 6. Linear regression analysis of prediction.

TABLE 3. Comparison with other models.

Prediction steps	Model	MAE	RMSE
1-day	Persistence	29.45	49.14
	LSTM	29.82	52.37
	EMD-LSTM	25.50	43.86
	CNN-LSTM	25.39	43.12
	SVR	29.60	50.06
	EMD-SVR	23.98	44.05
	EMD-CNN-LSTM	**17.61**	**35.04**
	SFM	19.98	**39.27**
1-week	Persistence	68.82	112.71
	LSTM	67.73	108.65
	EMD-LSTM	38.03	63.68
	CNN-LSTM	44.62	74.51
	SVR	68.09	107.26
	EMD-SVR	36.94	61.07
	EMD-CNN-LSTM	**19.88**	**39.60**
	SFM	27.53	46.90

transform. The number of hidden units are set to 256 and 32 in these two models. The CNN-LSTM extracts the features directly from original sequence through CNN and models the time dependencies of these features through LSTM. The length of convolutional kernels is set to 64 and the number of hidden units of LSTM is set to 32.

To compare the performance with other prediction models for time series, SVR and EMD-SVR are also included in our experiments. As a widely applied model for predicting time series, SVR maps the input data to a high-dimensional space, and then uses linear regression on features in high-dimensional space to fit the data in original space [23]. Radial basis function (RBF) is used as kernel function, epsilon is set to 0.01, C is set to 1.0, and gamma is set to 0.01. The results of models are presented in table 3. MAE represents the prediction error of these models. Both in one-day and one-week prediction, the proposed model gets the smallest MAE among the models. The performance of LSTM and SVR, is close to persistence method. Combined with EMD, the performance of these two networks has been improved significantly. We can also see from table 3 that the prediction error is significantly reduced by EMD-CNN-LSTM compared with CNN-LSTM and EMD-LSTM.

Finally, linear regression is used to assess the predictive accuracy of the model and the results are shown in figure 6. The determination coefficient R^2 is used to test the degree of association between the two variables. In 1-day prediction, $R^2 = 0.9913$ and in 1-week prediction $R^2 = 0.9901$, which are close to 1. This means that our model can predict the stock price accurately.

V. CONCLUSION

In this paper, we build a financial series prediction model by combining EMD, CNN and LSTM to predict stock price. The model uses 10-minute samples of SSE to predict daily and weekly stock price. The main steps are as follow: (1) divide every 1000 consecutive sampling points into 4 sub-series; (2) decompose each sub-series into five components representing five levels of frequency features; (3) extracting frequency features through a CNN for each component; (4) combine all frequency features by a fully connected layer and extract the time dependencies of frequency features from four sub-series; (5) get the final predicted price after a linear transform of LSTM outputs. Compared with several state-of-art models, the proposed model has been verified effective. In addition, we study the impact of frequency information on prediction performance. The conclusions are summarized as follow:

- Different frequency components represent features at different time scales. Longer convolutional kernels are suitable for low frequency component to extract features at long time scales, and vice versa.
- The highest frequency contains amount of noise as well as a number of features at extremely short time scales. For longer step prediction, remove these components can improve prediction accuracy.

- Residue, the lowest frequency component of a stock price series, represent most effective features which are of significance to the prediction performance.

There are some points to further study to improve the model. First, all the features in this paper are extracted from raw series, instead of taking financial characteristics into account. Combining frequency features and financial characteristics, like volume, highest price, lowest price and so on, in appropriate time scales, may be a further study direction. Second, there are some improved versions, such as ensemble empirical mode decomposition (EEMD) and complementary ensemble empirical mode decomposition (CEEMD). Whether the improved versions of EMD can improve prediction performance is not clear. Finally, how to improve the model to predict other nonlinear and non- stationary series, such as wind speed, traffic, earthquake wave and traffic flow. Finally, the idea of EMD is similar to the wavelet decomposition. EMD is a frequency transformation and decomposes original series into multiple independent vectors, while wavelet transformation is a time-frequency transformation and decomposes the series into a two-dimensional matrix (time and frequency). Price prediction based on time-frequency features can be our future work.

REFERENCES

[1] J. Cao, Z. Li, and J. Li, "Financial time series forecasting model based on CEEMDAN and LSTM," *Phys. A, Stat. Mech. Appl.*, vol. 519, pp. 127–139, Apr. 2019.
[2] F. T. Magiera, "Forecasting volatility in financial markets: A review," *CFA Dig.*, vol. 34, no. 1, pp. 84–85, Feb. 2004.
[3] Y. Hu, B. Feng, X. Zhang, E. W. T. Ngai, and M. Liu, "Stock trading rule discovery with an evolutionary trend following model," *Expert Syst. Appl.*, vol. 42, no. 1, pp. 212–222, Jan. 2015.
[4] W. Nuij, V. Milea, F. Hogenboom, F. Frasincar, and U. Kaymak, "An automated framework for incorporating news into stock trading strategies," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 4, pp. 823–835, Apr. 2014.
[5] Q. Gao, "Stock market forecasting using recurrent neural network," Ph.D. dissertation, Dept. Comput. Eng., Univ. Missouri, Columbia, MO, USA, 2016.
[6] K.-J. Kim and H. Ahn, "Simultaneous optimization of artificial neural networks for financial forecasting," *Int. J. Speech Technol.*, vol. 36, no. 4, pp. 887–898, Jun. 2012.
[7] K.-J. Kim and I. Han, "Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index," *Expert Syst. Appl.*, vol. 19, no. 2, pp. 125–132, Aug. 2000.
[8] A. A. Adebiyi, A. O. Adewumi, and C. K. Ayo, "Comparison of ARIMA and artificial neural networks models for stock price prediction," *J. Appl. Math.*, vol. 2014, pp. 1–7, 2014.
[9] Y. Wang, L. Zhang, Y. Liu, and J. Guo, "Gold price prediction method based on improved PSO-BP," *Int. J. Service, Sci. Technol.*, vol. 8, no. 11, pp. 253–260, Nov. 2015.
[10] T. Kuremoto, S. Kimura, K. Kobayashi, and M. Obayashi, "Time series forecasting using a deep belief network with restricted Boltzmann machines," *Neurocomputing*, vol. 137, pp. 47–56, Aug. 2014.
[11] R. Xiong, E. P. Nichols, and Y. Shen, "Deep learning stock volatility with Google domestic trends," 2015, *arXiv:1512.04916*. [Online]. Available: http://arxiv.org/abs/1512.04916
[12] A. M. Saxe, J. L. McClelland, and S. Ganguli, "Exact solutions to the nonlinear dynamics of learning in deep linear neural networks," 2013, *arXiv:1312.6120*. [Online]. Available: http://arxiv.org/abs/1312.6120
[13] L. Zhang, C. Aggarwal, and G.-J. Qi, "Stock price prediction via discovering multi-frequency trading patterns," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2017, pp. 1–5.
[14] N. E. Huang, Z. Shen, S. Long, M. Wu, and H. Shih, "The empirical mode decomposition and Hilbert spectrum for nonlinear and nonstationary time series analysis," *Proc. Roy. Soc. Math. Phys. Eng. Sci.*, vol. 454, pp. 903–995, Mar. 1998.
[15] L. Yu, S. Wang, and K. K. Lai, "Forecasting crude oil price with an EMD-based neural network ensemble learning paradigm," *Energy Econ.*, vol. 30, no. 5, pp. 2623–2635, Sep. 2008.
[16] Y. Ren, P. N. Suganthan, and N. Srikanth, "A comparative study of empirical mode decomposition-based short-term wind speed forecasting methods," *IEEE Trans. Sustain. Energy*, vol. 6, no. 1, pp. 236–244, Jan. 2015.
[17] G. Rilling, P. Flandrin, and P. Gonalves, "On empirical mode decomposition and its algorithms," in *Proc. IEEE-EURASIP Workshop Nonlinear Signal Image Process. (NSIP)*, Jun.-2013, pp. 8–11.
[18] S. Abdoli, P. Cardinal, and A. Lameiras Koerich, "End-to-end environmental sound classification using a 1D convolutional neural network," *Expert Syst. Appl.*, vol. 136, pp. 252–263, Dec. 2019.
[19] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[20] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," *Proc. Int. Conf. Learn. Represent.*, pp. 1–41, 2015.
[21] J. Hu, J. Wang, and G. Zeng, "A hybrid forecasting approach applied to wind speed time series," *Renew. Energy*, vol. 60, pp. 185–194, Dec. 2013.
[22] S. Makridakis, "Accuracy measures: Theoretical and practical concerns," *Int. J. Forecasting*, vol. 9, no. 4, pp. 527–529, Dec. 1993.
[23] N. Sapankevych and R. Sankar, "Time series prediction using support vector machines: A survey," *IEEE Comput. Intell. Mag.*, vol. 4, no. 2, pp. 24–38, May 2009.
[24] M. Seo, S. Lee, and G. Kim, "Forecasting the volatility of stock market index using the hybrid models with Google domestic trends," *Fluctuation Noise Lett.*, vol. 18, no. 01, Mar. 2019, Art. no. 1950006.
[25] Z. Zeng, H. Xiao, and X. Zhang, "Self CNN-based time series stream forecasting," *Electron. Lett.*, vol. 52, no. 22, pp. 1857–1858, Oct. 2016.
[26] A. M. Awajan and M. T. Ismail, "A hybrid approach EMD-HW for short-term forecasting of daily stock market time series data," *AIP Conf. Process.*, vol. 1870, no. 1, 2017, Art. no. 060006, doi: 10.1063/1.4995933.
[27] T. Li, M. Hua, and X. Wu, "A hybrid CNN-LSTM model for forecasting particulate matter (PM2.5)," *IEEE Access*, vol. 8, pp. 26933–26940, 2020, doi: 10.1109/access.2020.2971348.
[28] A. Vidal and W. Kristjanpoller, "Gold volatility prediction using a CNN-LSTM approach," *Expert Syst. Appl.*, vol. 157, Nov. 2020, Art. no. 113481, doi: 10.1016/j.eswa.2020.113481.
[29] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2015, pp. 4580–4584, doi: 10.1109/icassp.2015.7178838.

WANGWEI SHU (Graduate Student Member, IEEE) received the B.E. degree in electronic information engineering from Beihang University, in 2018. He is currently pursuing the master's degree with Beihang University. He has contributed to the theoretical trading strategies and engineering realization of quantitative trading in his laboratory. His current research interests include quantitative trading, temporal signal processing, and deep learning.

QIANG GAO (Senior Member, IEEE) received the Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Sciences, in 2000. From 2001 to 2004, he worked with the German National Information Technology Research Center and Aston University, U.K. He is currently a Professor with Beihang University. He has published more than 30 articles in the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS (TWC), the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY (TVT), *Electronics Letters*, *Wireless Networks* (ACM), and other international top journals and conferences. He has authorized 15 invention patents, including two international patents and two software copyrights. His current research interests include time series analysis, deep learning, and wireless communication.