Accepted Manuscript

Financial time series forecasting model based on CEEMDAN and LSTM

Jian Cao, Zhi Li, Jian Li

PII: S0378-4371(18)31498-5
DOI: https://doi.org/10.1016/j.physa.2018.11.061
Reference: PHYSA 20379

To appear in: *Physica A*

Received date : 7 February 2018
Revised date : 14 August 2018

Please cite this article as: J. Cao, Z. Li and J. Li, Financial time series forecasting model based on CEEMDAN and LSTM, *Physica A* (2018), https://doi.org/10.1016/j.physa.2018.11.061

Highlights:

●A new hybrid time series forecasting method is established by combining EMD and CEEMDAN algorithm with LSTM neural network.

● The forecasting efficiency of financial time series is improved by the model.

● The forecasting results of the proposed model are more accurate than other similar models.

Financial time series forecasting model based on CEEMDAN and LSTM

Jian Cao, Zhi Li*, Jian Li

School of Electronic and Information Engineering, Sichuan University, Chengdu 610065, China.

Abstract:

In order to improve the accuracy of the stock market prices forecasting, two hybrid forecasting models are proposed in this paper which combine the two kinds of empirical mode decomposition (EMD) with the long short-term memory (LSTM). The financial time series is a kind of non-linear and non-stationary random signal, which can be decomposed into several intrinsic mode functions of different time scales by the original EMD and the complete ensemble empirical mode decomposition with adaptive noise (CEEMDAN). To ensure the effect of historical data on the prediction result, the LSTM prediction models are established for all each characteristic series from EMD and CEEMDAN deposition. The final prediction results are obtained by reconstructing each prediction series. The forecasting performance of the proposed models is verified by linear regression analysis of the major global stock market indices. Compared with single LSTM model, support vector machine (SVM), multi-layer perceptron (MLP) and other hybrid models, the experimental results show that the proposed models display a better performance in one-step-ahead forecasting of financial time series.

Keywords: Financial time series forecasting; EMD-LSTM prediction; CEEMDAN-LSTM prediction

1. Introduction

Stock market price forecast is an important issue to the professional researchers and investors[1]. In recent years, as an auxiliary tool for the prediction of financial time series, ANN has a good performance[4]. Some ANN, like the back propagation (BP) neural networks[9], fit multi-parameter non-linear functions through adaptive learning, and obtain good clustering ability[10]. Due to ANN's data-driven characteristic, the future prices is predicted through the historical and current time data. Mehdi Khashei et al.[11] combined fuzzy models, autoregressive integrated moving average (ARIMA) with BP neural networks to predict the financial time series. Ratnadip Adhikari et al.[12] established the financial time series prediction model of random walk and ANN. Kuremoto T et al.[13] predicted the financial time series using deep belief networks and Boltzmann machines. Wang et al.[14] applied EMD algorithm and stochastic time neural networks to forecast financial time series. Additionally, SVM has also been proved to be an effective time series prediction model[15].

The change of stock price is non-linear and non-stationary. So it is quite difficult to predict price fluctuation reliably and accurately. Since the price prediction in stock market is not only related to the data at the current time but also to the data at an earlier time, the information carried by the data at an earlier time will be lost if only the data at the latest time is applied. Unlike traditional ANNs, recurrent neural networks (RNN) establishes connections between hidden units, which makes this network enable to keep memory of recent events[18]. RNN deals with the before-after associated data by the memory characteristics. It is very suitable for the prediction of time series[19][20]. As an improved model of RNN, LSTM is widely used in natural language recognition, time series prediction and other

* Corresponding author.
E-mail address: lizhi@scu.edu.cn (Z. Li).

fields[21]. The information is selectively filtered through a "gate" structure of LSTM, and more useful information is extracted from historical data in training.

In order to reduce the impact of noise on the prediction, EMD and its advanced version, CEEMDAN, are combined with LSTM to predict the financial time series. As EMD is a Fourier transform-based signal decomposition method, it processes any non-linear and non-stationary signal adaptively. The original time series is decomposed into several sub-series under different frequencies by EMD processing, and these sub-series are respectively predicted as the input data of LSTM model. Finally, all the predictions are reconstructed to get the final result. In Section 3 of this paper, linear regression is used to assess the predictive accuracy of the model. The original data in this paper comes from four major global stock indices, including Standard & Poor 500 index (S&P500), the Hang Seng Index (HSI), the Deutscher Aktien Index (DAX) and the Shanghai Stock Exchange Composite Index (SSE).

2. Methodology

2.1 EMD, CEEMDAN algorithm

In order to solve the difficulty to deal with random data in ANNs, the original non-linear and non-stationary financial time series $S(t)$ is preprocessed.

2.1.1 EMD

Empirical mode decomposition is an adaptive signal time-frequency processing method proposed by Huang et al.[23], which decomposes the signal according to the time scale feature of the data itself without pre-setting of any basis function. It decomposes the series into a finite number of intrinsic mode functions (IMFs). Each IMF component represents different feature of the original signal at different time scales. EMD decomposition process is called the screening process, the steps are the following: (1) Find all the maxima and minima in the time series $S(t)$. (2) All the extremum points are fitted to the upper envelope $U(t)$ and the lower envelope $L(t)$ by the cubic spline interpolation function. (3) Calculate the mean of the upper envelope and the lower envelope by $m(t) = (U(t) + L(t))/2$. (4) Subtract $m(t)$ from the original series $S(t)$ gives a new series $h(t) = S(t) - m(t)$. (5) Repeat steps (1) through (4) with $h(t)$ as a new input series until the mean of $h(t)$ approaches zero, get the ith IMF, denoted $C_i(t)$, i as its index. (6) $C_i(t)$ is separated from the original series $S(t)$ to get a difference series without high frequency components $r_i(t) = S(t) - C_i(t)$. (7) The steps (1) to (6) are repeated with $r_i(t)$ as the new input series until the termination condition is satisfied (typically such that the last residue satisfies monotonicity). Through the above steps, we screened a series of IMFs, recorded as $C_i(t)$ (i=1,2,...,N). The original signal $S(t)$ is reconstructed by these IMFs.

$$R_N(t) = S(t) - \sum_{i=1}^{N} C_i(t) \tag{1}$$

Where $R_N(t)$ is the residue, representing the trend of the time series.

2.1.2 CEEMDAN

The EMD has great advantages in dealing with non-stationary and non-linear signals, but it still has "mode mixing" problem. Mode mixing refers to the presence of very similar oscillations in different modes or very disparate amplitude in a mode. By adding Gaussian white noise to the signal, the ensemble empirical mode decomposition (EEMD) algorithm largely eliminates the mode mixing in

EMD algorithm[24]. However, the EEMD algorithm cannot completely eliminate Gaussian white noise after signal reconstruction, it cause reconstruction errors. To solve these problems, the complete ensemble empirical mode decomposition with adaptive noise (CEEMDAN) was proposed as an improved version of EEMD[25]. It eliminate mode mixing more effectively, the reconstruction error is almost zero, and the cost of calculation is greatly reduced. Define the operator $E_j(\cdot)$ that produces the *j*th mode obtained by EMD, and let $w_i(t)$ be white noise with normal distribution $N(0,1)$. The steps of CEEMDAN algorithm are the following: (1) Decompose each $S_i(t)=S(t)+\varepsilon_0 w_i(t)$ by EMD to extract the first IMF, where ε_0 is a noise coefficient, *i*=1,2,...*I*, and define the first mode as $\overline{IMF_1}=\frac{1}{I}\sum_{i=1}^{I} IMF_{i1}$. (2) Calculate the first residue $r_1(t)=S(t)-\overline{IMF_1}$. (3) Decompose residue $r_1(t)+\varepsilon_1 E_1(w_i(t))$ to obtain the second mode as $\overline{IMF_2}=\frac{1}{I}\sum_{i=1}^{I} E_1(r_1(t)+\varepsilon_1 E_1(w_i(t)))$. (4) Repeat for another IMF until the obtained residue. The final residue can be expressed as:

$$R_M(t)=S(t)-\sum_{j=1}^{M}\overline{IMF_j} \tag{2}$$

Where *M* is the total number of IMFs. The IMFs together compose the characteristics of the original signal at different timescales. The residue clearly shows the trend of the original sequence, which is smoother and reduce the prediction error effectively.

2.2 Forecasting model

2.2.1 Long short-term memory

The one-step-ahead prediction of the financial time series requires not only the latest data, but also the previous data. Benefit of the self feedback mechanism of the hidden layer, the RNN model has an advantage in deal with long-term dependence problems, but there are difficulties in practical application[26]. To solve the problem of gradient disappearance of RNN, Sepp Hochreiter and Jurgen Schmidhuber proposed the LSTM model in 1997[27], it was recently improved and promoted by Alex Graves[28]. LSTM unit consists of a memory cell that stores information which is updated by three special gates : the input gate, the forget gate and the output gate. LSTM unit structure shown in Fig. 1.

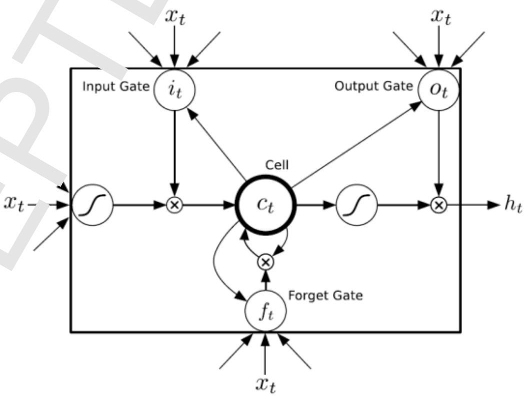


Fig.1. LSTM unit structure [28]

At time t, x_t is the input data of the LSTM cell, h_{t-1} is the output of the LSTM cell at the previous moment, c_t is the value of the memory cell, h_t is the output of the LSTM cell. The calculation process of LSTM unit can be divided into the following steps.

(1) First, calculate the value of the candidate memory cell $\tilde{c}_t$, W_c is the weight matrix, b_c is the bias.

$$\tilde{c}_t=\tanh(W_c\cdot[h_{t-1},x_t]+b_c) \tag{3}$$

(2) Calculate the value of the input gate i_t, the input gate controls the update of the current input data to the state value of the memory cell, σ is sigmoid function, W_i is the weight matrix, b_i is the bias.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{4}$$

(3) Calculate the value of the forget gate f_t, the forget gate controls the update of the historical data to the state value of the memory cell, W_f is the weight matrix, b_f is the bias.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{5}$$

(4) Calculate the value of the current moment memory cell c_t, and c_{t-1} is the state value of the last LSTM unit.

$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t \tag{6}$$

Where "*" represents dot product. The update of memory cell depends on the state value of the last cell and the candidate cell, and it is controlled by input gate and forget gate.

(5) Calculate the value of the output gate o_t, the output gate controls the output of the state value of the memory cell, W_o is the weight matrix, b_o is the bias.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{7}$$

(6)Finally, calculate the output of LSTM unit h_t.

$$h_t = o_t * \tanh(c_t) \tag{8}$$

Benefit from three control gates and memory cell, LSTM keep, read, reset and update long time information easily. It is important to note that, due to the sharing mechanism of the LSTM internal parameters, the dimensions of the output can be controlled by setting the dimensions of the weight matrix. LSTM establishes a long time delay between input and feedback. The gradient will neither explode nor disappear because the internal state of the memory cell in this architecture maintains a continuous error flow.

2.2.2 Dropout

Preventing over-fitting is important during training. Dropout was put forward to prevent over-fitting by Hinton[29]. During every training process of network, some units are randomly discarded from a network at a certain probability. As shown in Fig.2, the left figure is a fully connected network, and the right figure is the network after Dropout is applied. It is equivalent to a relatively smaller network each time.

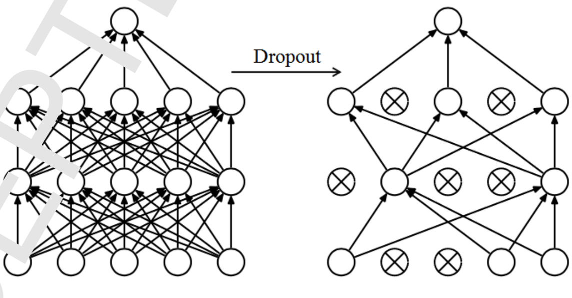


Fig.2. Flow chart of Dropout [29]

Yarin Gal proposed the application of dropout in RNN[30]. In this dropout variant, we repeat the same dropout mask at each time step for both inputs, outputs, and recurrent layers (drop the same network units at each time step). The technique of dropout applied to RNN is shown in Fig. 3, where each square represents an RNN unit, horizontal arrows represent the recurrent connections, and vertical arrows represent the input and output of each RNN unit. The same color connection corresponds to the same dropout mask. The method of applying dropout in LSTM is the same as that of RNN. It is noteworthy that, in each training, the units discarded are random, the parameters of these units will not be updated, and all the units will be used in the testing process.

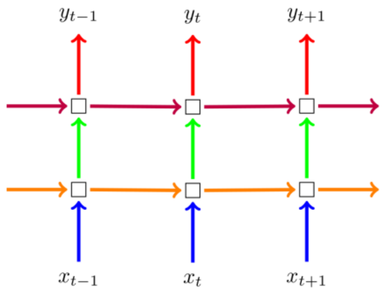


Fig.3. Dropout technique in RNN [30]

2.3 Proposed method

To establish a reliable stock market forecasting model, two layers of LSTM are used in this study. Because the output of the LSTM unit is a multidimensional vector, the output of the second layer of the last LSTM unit is connected to a fully connected layer. Finally, a prediction value is output in layer4. The network structure of the proposed model is shown by Fig. 4.

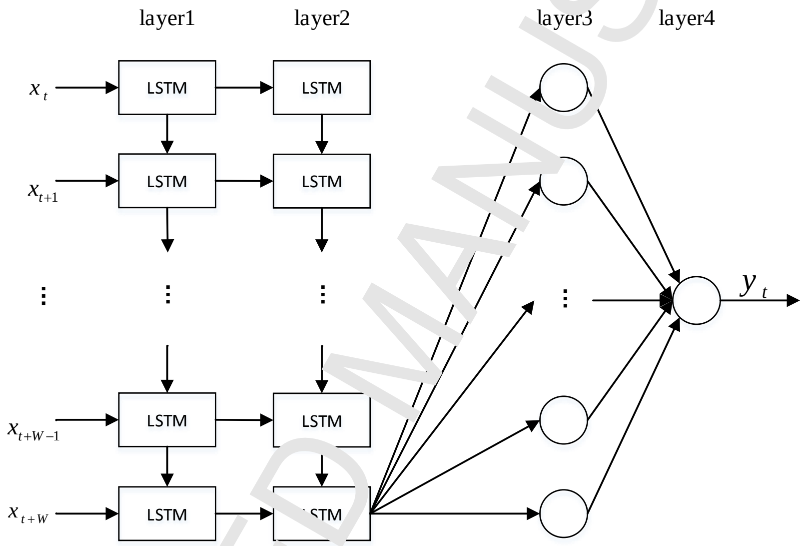


Fig.4. The network structure of the proposed model

Because of the supervised learning method, the input time series is converted to one sample with one label. The length of the training set time series is T, the window size is W, the input of the tth sample is (x_t , x_{t+1} ,..., x_{t+W}), and the corresponding label is x_{t+W+1} . In this way, the number of samples is $N = T - W$. In order to extract more features, the one dimensional value of the input is converted into a vector of *dim1* dimension in the first layer, we let *dim1*=128. In the second layer, the input vector of *dim1* dimensional is converted into a vector of *dim2* dimensional, we let *dim2* = 64. To get the predicted value, only the output v of the last cell is taken from the second layer. At this point v is a vector as the full connection layer input data, To further extract features, layer3 has 16 units, the last layer has only one unit, it will output a predicted value y. The activation function of all fully connected units is the "Relu" function. We use the mean square error (MSE) as the loss function.

$$Loss = MSE = \frac{1}{N}\sum_{n=1}^{N}(d_n - y_n)^2 \tag{9}$$

Where d_n is the original value corresponding to the nth sample and y_n is the predicted value. In multiple training iterations, the gradient descent method is used to reduce the loss function. The adam optimization algorithm is used to speed up learning in training[31]. This model is based on Keras deep learning framework. The implementation steps of the proposed model are shown in Fig.5.

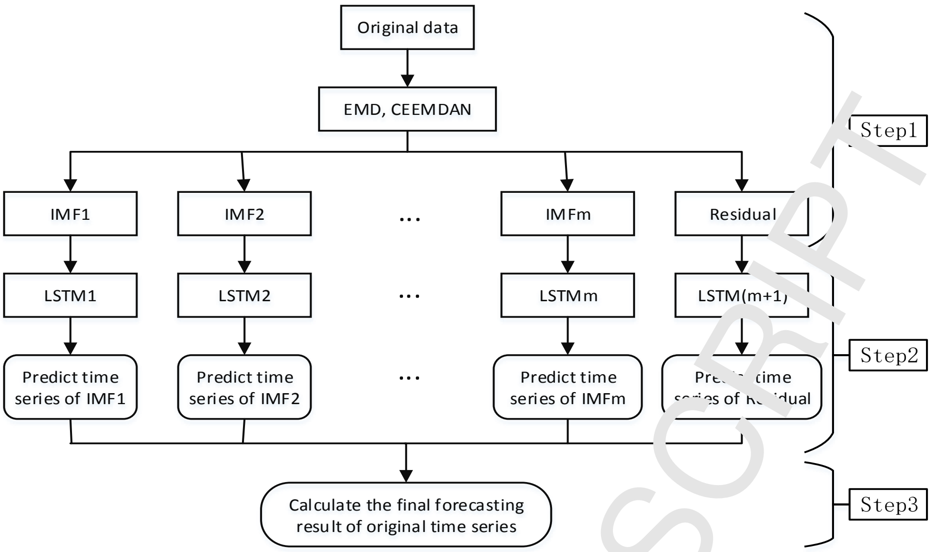


Fig.5. Flow chat of the proposed model

Step 1. Firstly, EMD and CEEMDAN are used to decompose the original financial time series $S(t)$ into several IMFs sequences $C_i(t)$ (i = 1, 2, ..., M) and a residue $R_M(t)$;

Step 2. The obtained IMFs and the residue are used as the input data of the LSTM forecasting model to train and obtain the predicted results respectively. The predicted result of the test set is $\tilde{C}_i(t)$ (i = 1, 2, ..., M) and $\tilde{R}_M(t)$.

Step 3. Finally, the prediction results obtained from each IMF and the residue are reconstructed according to the following formula to obtain the final predicted time series.

$$\tilde{S}(t) = \sum_{i=1}^{M} \tilde{C}_i(t) + \tilde{R}_M(t) \quad (t = 1, 2, ..., L) \tag{10}$$

Where L is the length of the test series, $\tilde{C}_i(t)$ is the prediction series of each IMF, $\tilde{R}_M(t)$ is the predictive series of the residue, and $\tilde{S}(t)$ is the final predictive series of the test set.

3. Analysis of experimental results

3.1 The processing of data

To test the performance of the proposed forecasting model, the daily closing price of the S&P500, HSI, DAX and SSE are selected as the original data, all of which are obtained from Yahoo Finance (https://finance.yahoo.com). Fig. 6 shows the original data of the financial time series which is volatile and non-stationary in short term. The statistical analysis of the original time series data is shown in Table 1. The data of all indices are from December 13, 2007 to December 12, 2017. Selecting the top 90% data of each time series as training set, and the latter 10% data as the test set. For a number of reasons, there are a small number of non-trading hours on the stock market. Data on these vacancies are excluded and only the data for the trading hours are kept for experimentation.

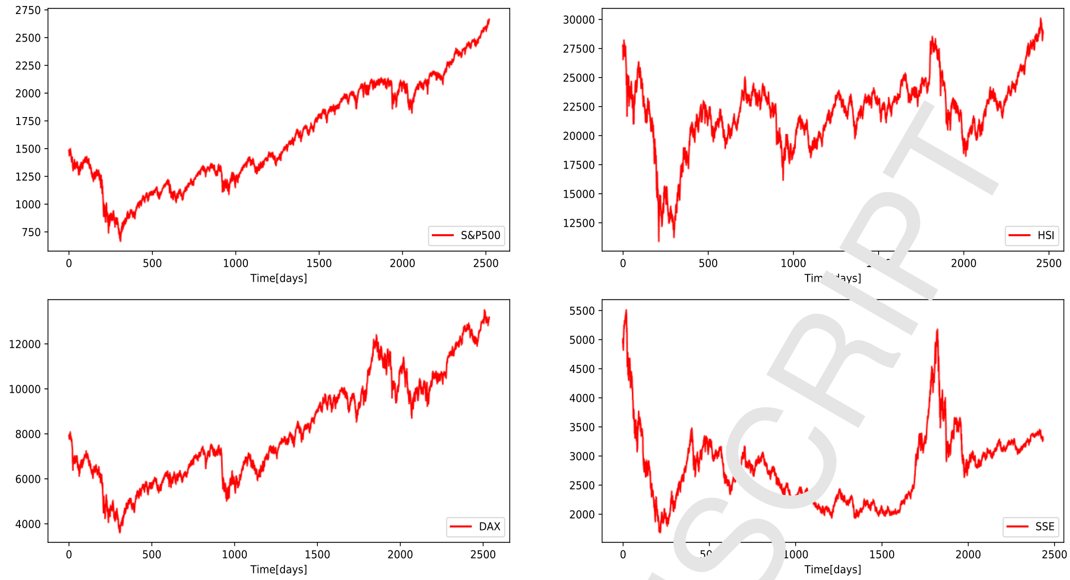


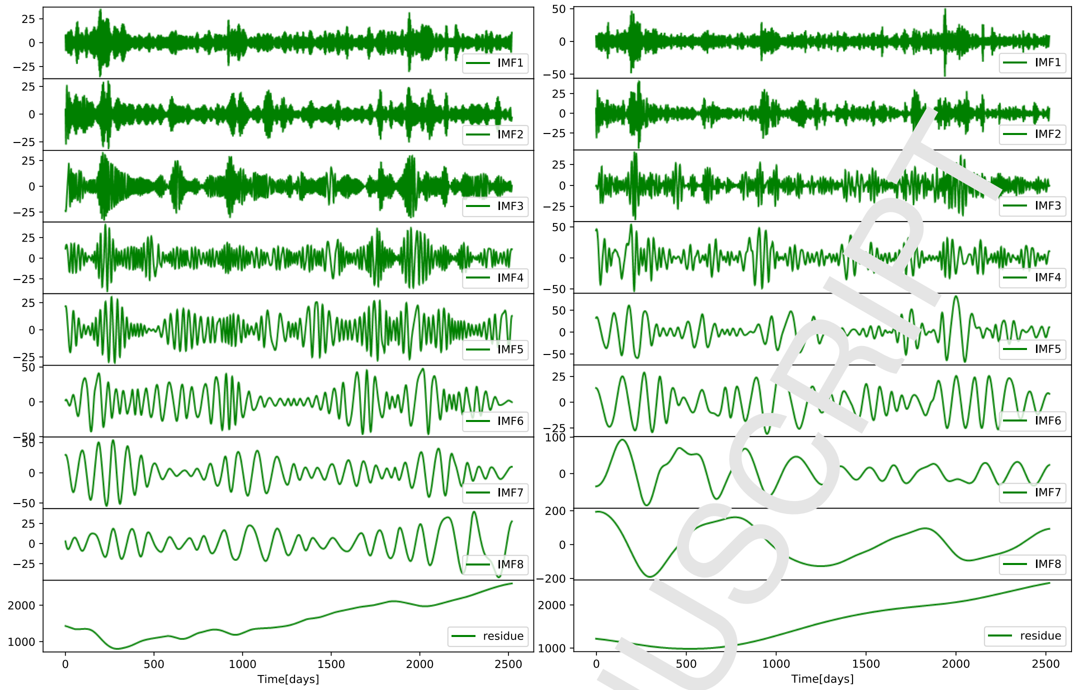
Fig.6. The original series of four stock daily closing price

Table 1 the statistical analysis results of four stock price data

Index	Count	Mean	Min	Max	Standard Deviation
S&P500	2518	1607.46	676.53	2664.11	477.02
HSI	2465	22000.22	[illegible]	30003.49	3068.58
DAX	2535	8198.87	[illegible]	13478.86	2372.72
SSE	2433	2802.20	1706.70	5497.90	648.06

3.2 EMD, CEEMDAN of financial time series

The original financial time series is decomposed into several IMFs and one residue through EMD and CEEMDAN. According to the experimental results, the number of IMFs generated by CEEMDAN is often less than the result generated by EMD algorithm. To compare the decomposition effect of the two algorithms and reduce the amount of computation, we limit the number of IMFs generated by EMD is the same as that of CEEMDAN. Fig. 7 shows the decomposition results of S&P500 index series. Each IMF is arranged from high frequency to low frequency. The first few IMFs represent high-frequency components or noises in the original series, though they are difficult to predict, but they are also important parts of the whole prediction. The decomposition results of EMD algorithm have "mode mixing" phenomenon, the IMFs obtained by CEEMDAN algorithm have obvious frequency difference.



(a) Decomposed by EMD (b) Decomposed by CEEMDAN

Fig.7. Decomposition results of S&P500 index

After decomposing each index time series by EMD algorithm, each data is normalized to the range of 0 to 1. it reduces the effects of noise, and ensures that neural networks update parameters efficiently and speed up training of the network[37]. We use the following formula for normalization.

$$x(t)' = \frac{x(t) - \min x(t)}{\max x(t) - \min x(t)} \tag{11}$$

Where min$x(t)$ and max$x(t)$ are the minimum and maximum value of the whole time series respectively. Since the data is normalized during the model training phase, the output of the test set can be restored by the formula $x(t)_p = x(t)'_p(\max x(t) - \min x(t)) + \min x(t)$, where $x(t)'_p$ is the output value of the forecasting model.

3.3 Training process and prediction results

After decomposition, each sub-series is divided into a training set and a test set, and then forecasting models are built for each sub-series. To obtain the best prediction result for different index data and different sub-series, the optimal hyper-parameters are selected through experiments, as shown in Table 2-3. The "window" represents the length of each sample series, and "epoch" is the number of training. As can be seen, the epoch of training for the residue is less than that of other IMFs, this is because the residue is a smooth curve, the model is easy to get the characteristics.

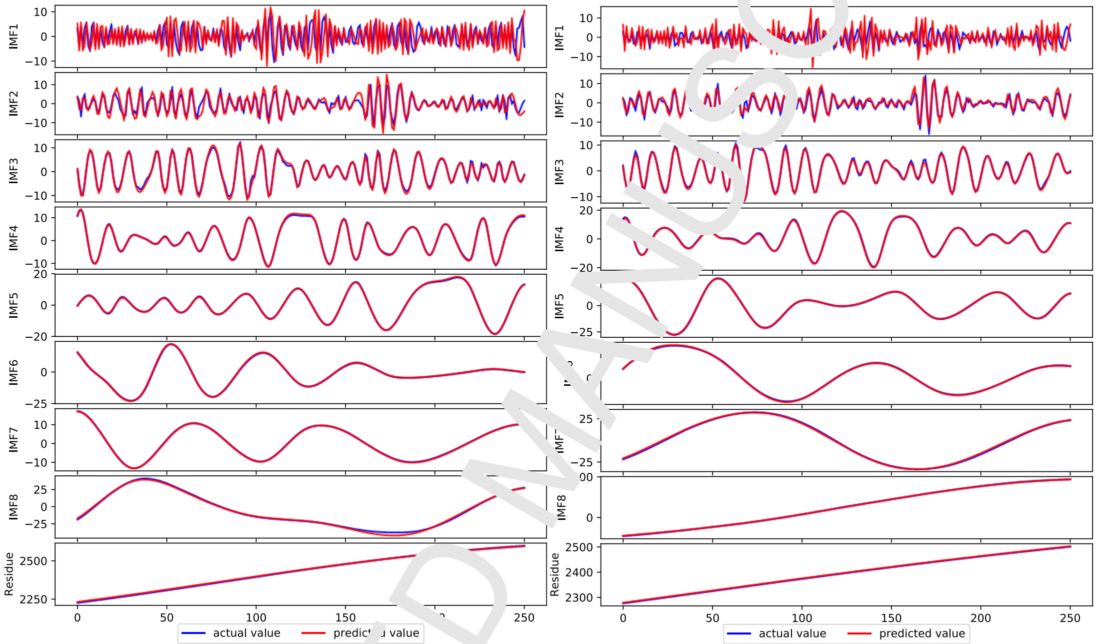
Table 2 Hyper-parameters of EMD-LSTM

	S&P500		HSI		DAX		SSE	
	window	epoch	window	epoch	window	epoch	window	epoch
IMF1	2	250	2	300	2	280	2	320
IMF2	[illegible]	200	2	300	2	260	3	280
IMF3	2	250	3	280	3	260	2	280
IMF4	3	260	3	280	3	200	2	260
IMF5	3	200	4	200	4	180	3	260
IMF6	4	160	4	160	5	200	5	200
IMF7	4	200	4	100	-	-	-	-
IMF8	4	100	4	90	-	-	-	-
residue	5	80	5	100	5	120	5	200

Table 3 Hyper-parameters of CEEMDAN-LSTM

	S&P500		HSI		DAX		SSE	
	window	epoch	window	epoch	window	epoch	window	epoch
IMF1	2	300	2	300	2	300	2	250
IMF2	2	200	2	200	2	240	2	150
IMF3	2	200	2	200	4	200	2	100
IMF4	4	200	2	200	4	200	2	100
IMF5	4	140	4	140	4	120	4	100
IMF6	4	150	4	120	4	60	4	100
IMF7	4	100	4	80	-	-	-	-
IMF8	4	60	4	50	-	-	-	-
residue	4	60	4	50	4	60	4	50

Fig.8 shows the forecast results of sub-series for S&P500 index test set data. The prediction accuracy of high frequency components IMF1 and IMF2 is relatively low due to the high amplitude and high frequency of the components. The residue represents the long-term trend of the index data, and the predicted value of the residue is close to the actual value.



(a) EMD-LSTM (b) CEEMDAN-LSTM

Fig.8 Forecast results of sub-series for S&P500 index

Finally, the forecast results of the four indices are shown in Fig.9 (a) (c) (e) (g). According to the pictures, The predictive values of the four financial time series are very close or even coincide with the original values. To assess the performance of the proposed models with more accuracy, three error measures are adopted, including mean absolute error (MAE), root mean square error (RMSE), and mean absolute percentage error (MAPE). They are calculated as follows:

$$MAE = \frac{1}{N}\sum_{t=1}^{N}\left|d_t - y_t\right| \tag{12}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{t=1}^{N}(d_t - y_t)^2} \tag{13}$$

$$MAPE = 100 \times \frac{1}{N}\sum_{t=1}^{N}\left|\frac{d_t - y_t}{d_t}\right| \tag{14}$$

In the above formula, d_t represents the original value of the t moment, y_t represents the predicted value of the t moment, and N is the total number of the test samples. If the value of MAE, RMSE, and MAPE is smaller, the smaller the deviation between the predicted value and the original value. The MAPE value is more stable than other error measure, so it is used as the main evaluating indicator. The prediction error of the two models is shown in Table 4. According to Table 4, The prediction error of CEEMDAN-LSTM to four indices is less than EMD-LSTM. Since the CEEMDAN extracts more effective features from original data, the prediction error is smaller.

Table 4 Prediction error of the two models

Index	model	MAE	RMSE	MAPE
S&P500	EMD-LSTM	5.0611	6.2794	0.2098
	CEEMDAN-LSTM	3.9177	4.8291	0.1617
HSI	EMD-LSTM	89.2689	118.9504	0.3418
	CEEMDAN-LSTM	71.8757	98.2410	0.2768
DAX	EMD-LSTM	36.1955	45.2663	0.2924
	CEEMDAN-LSTM	24.8473	33.3531	0.2010
SSE	EMD-LSTM	8.3783	10.8478	0.2587
	CEEMDAN-LSTM	6.8621	8.7431	0.2116

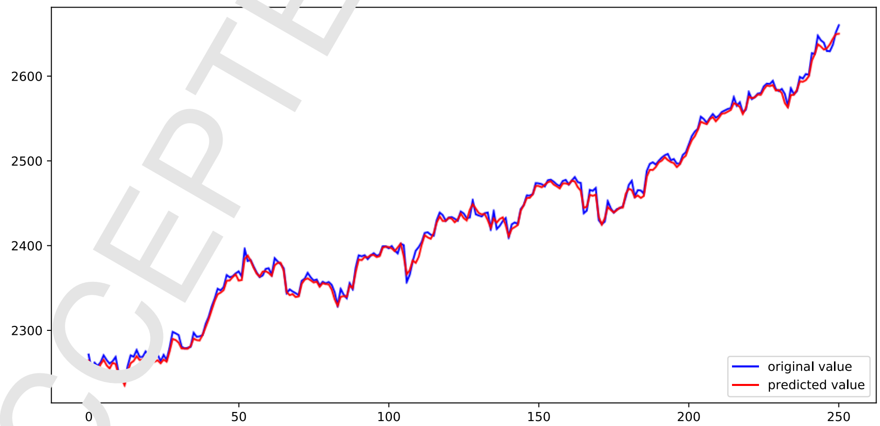
To better explain the performance of the forecasting model, the predictive value of the CEEMDAN-LSTM model and the original value are analyzed by linear regression. The prediction value is X, the original value is Y, then the linear equation is $Y=aX+b$. The determination coefficient R^2 is used to test the degree of association between the two variables. The R^2 is defined as follows:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - p_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})} \tag{15}$$

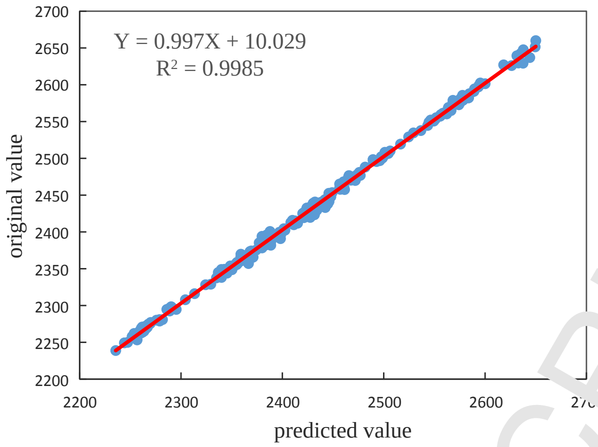
Where y is the original value, p is the predicted value, $\bar{y}$ is the average of the original values, and n is the number of test sample sets. If a is close to 1, the smaller the deviation between the predicted value and the original value. The linear regression of the experimental results is shown in Fig.9 (b) (d) (f) (h). The parameters in the linear regression analysis are also listed in Table 5. The slope a of the linear regression of each index is close to 1, and the determination coefficient R^2 is also close to 1, which shows that the predicted value of this model is very close to the original value.

Table 5 The parameters in the linear regression analysis

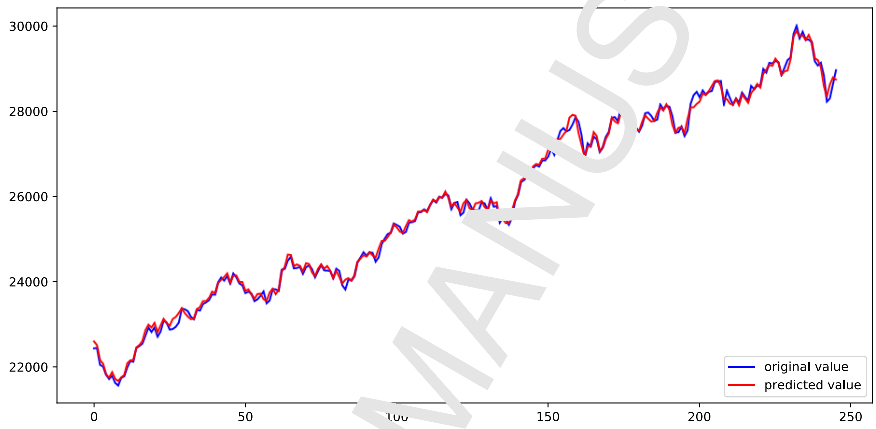
parameter	S&P500	HSI	DAX	SSE
a	0.997	1.0115	1.0007	0.998
b	10.029	-306.57	-12.211	8.8263
R^2	0.9985	0.9982	0.996	0.9932



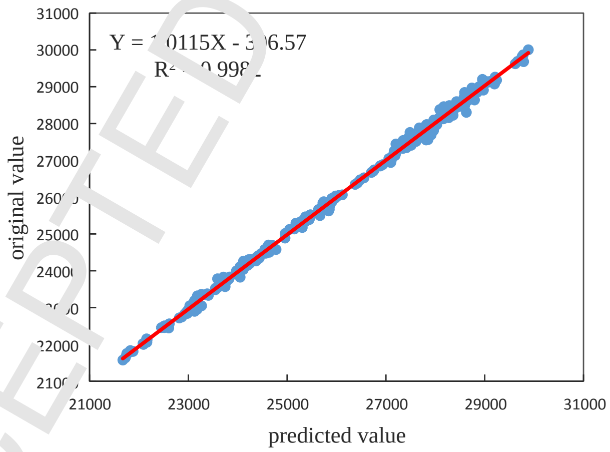
(a) Forecasting result of S&P500



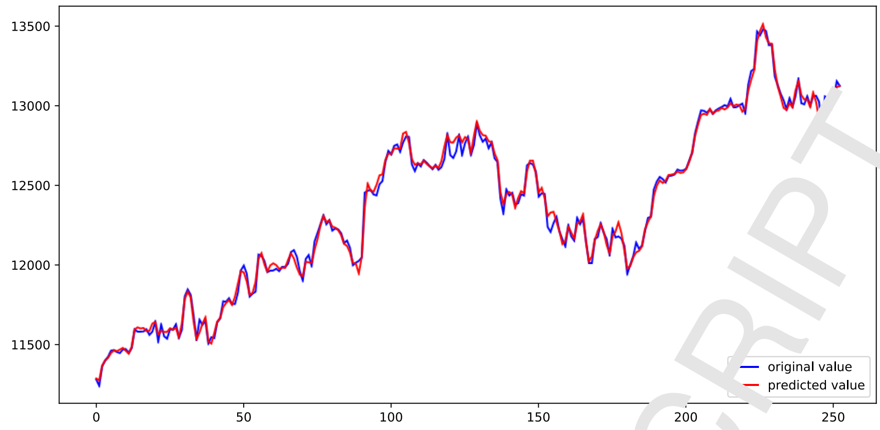
(b) Linear regression analysis ofS&P500



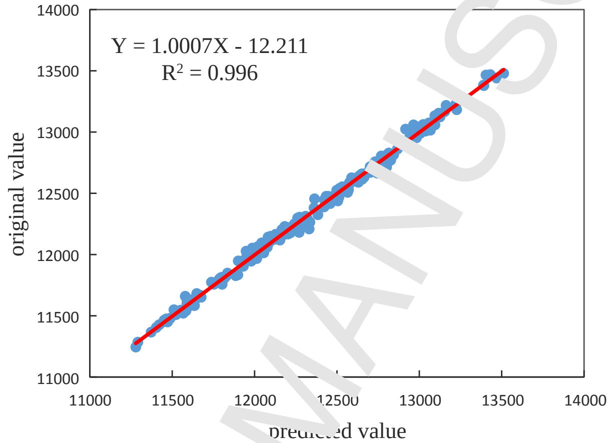
(c) Forecasting result of HSI



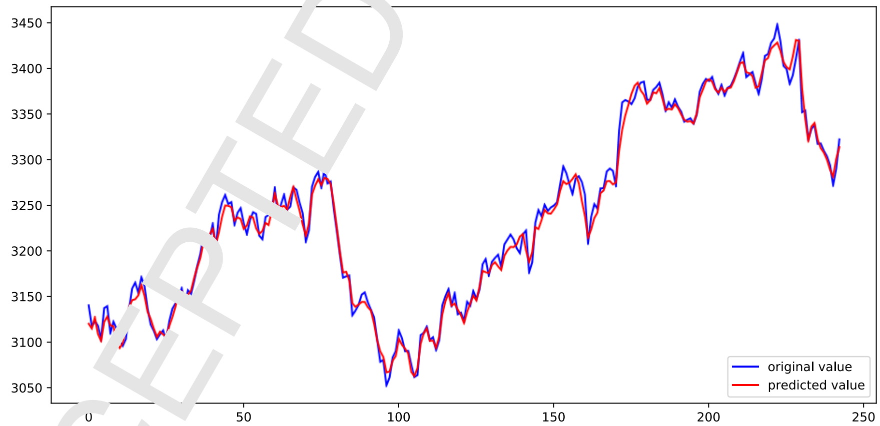
(d) Linear regression analysis of HSI



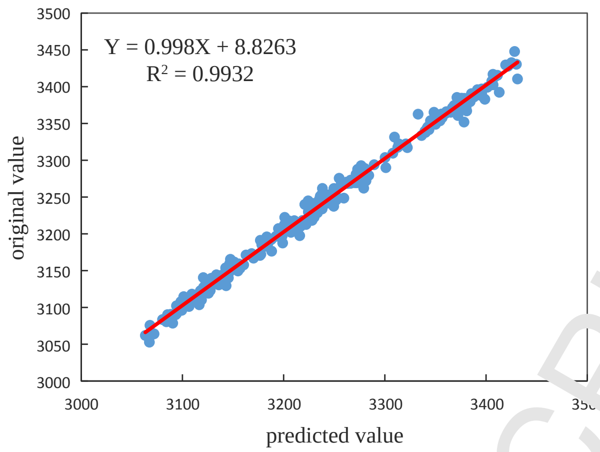
(e) Forecasting result of DAX



(f) Linear regression analysis of DAX



(g) Forecasting result of SSE



(h) Linear regression analysis of SSE

Fig.9. The forecast results of four series by CEEMDAN-LSTM

3.4 Comparison with other models

In this section, to verify the performance of the proposed method, we compared the prediction effect of several models, including LSTM, SVM, CEEMDAN-SVM, CEEMDAN-MLP, and CEEMDAN-LSTM. All models use the same data set. The original data is directly used as input data for the LSTM and SVM forecasting models. The other three hybrid models use the CEEMDAN algorithm with series decomposition. SVM has been widely applied to time series prediction[16]. SVM maps the input features to high-dimensional space, and uses the linear regression model in this high-dimensional space, so as to achieve the prediction of nonlinear data. The SVM model applied in this paper is implemented through the "scikit-learn" machine learning library. We use the commonly used radial basis function (RBF) as the kernel function[15]. MLP has a good generalization ability and widely used in the field of classification and prediction[4]. In this paper, we have optimized the various hyper-parameters of the MLP through experiments, and use a 4-layer network, one input layer, two hidden layers, and one output layer. Table 6 shows the prediction measures of the S&P500, HSI, DAX and SSE indices using these models. The error measures MAE, RMSE, and MAPE values of S&P500 index by the CEEMDAN-LSTM model are 3.9177, 4.8291 and 0.1617 respectively, which is much smaller than that of other models. The CEEMDAN-LSTM also performs better than the other models in predicting the HSI, DAX and SSE indices, and the predicted value is closer to the original value.

Table 6 Comparison of prediction measures of different models

Index	model	MAE	RMSE	MAPE
S&P500	LSTM	14.8064	18.2321	0.6115
	SVM	7.2209	10.2448	0.3993
	CEEMDAN-SVM	5.4011	6.8118	0.2213
	CEEMDAN-MLP	5.0199	6.1412	0.2051
	CEEMDAN-LSTM	**3.9177**	**4.8291**	**0.1617**
HSI	LSTM	169.0257	215.4280	0.6472
	SVM	147.6551	192.2751	0.5678
	CEEMDAN-SVM	102.6629	129.8097	0.3984
	CEEMDAN-MLP	89.7364	114.6355	0.3416
	CEEMDAN-LSTM	**71.8757**	**98.2410**	**0.2768**
DAX	LSTM	83.7904	109.5599	0.6731
	SVM	57.7602	80.8611	0.4678
	CEEMDAN-SVM	37.5861	49.0196	0.3043
	CEEMDAN-MLP	48.9322	58.3381	0.3981
	CEEMDAN-LSTM	**24.8473**	**33.3531**	**0.2010**
SSE	LSTM	14.8829	18.6694	0.4586
	SVM	17.4754	21.5559	0.5385
	CEEMDAN-SVM	10.1280	12.7900	0.3128
	CEEMDAN-MLP	10.3604	12.5282	0.3192
	CEEMDAN-LSTM	**6.8621**	**8.7431**	**0.2116**

4. Conclusion

In this study, the financial time series forecasting model (CEEMDAN-LSTM) is established by combining CEEMDAN signal decomposition algorithm with LSTM model. The model is applied to forecast the daily closing prices of major global stock indices (S&P500, HSI, DAX, SSE). The main steps are the following: (1) the stock index time series is decomposed into several intrinsic mode function and one residue by CEEMDAN; (2) LSTM prediction model is adopted for each IMF and the residue to get all the forecast results; (3) the final forecast result is obtained by restructuring the forecast results of each IMF and the residue. By error analysis (MAE, RMSE, MAPE), CEEMDAN-LSTM model shows the lower prediction error than other prediction models (such as LSTM, SVM, CEEMDAN-SVM, CEEMDAN-MLP, EMD-LSTM). Although the proposed hybrid models have a satisfied performance in the forecasting, there are some places to improve. For example, the daily closing price is only adopted as input data. To improve the accuracy and the robustness, more financial parameter, like the trading volume and the highest price, and the different time-scale series may be the input. In addition, the time series prediction method presented in this paper is also able to predict the other time series, such as the weather and traffic.

Conflict of Interests

The authors declare that they have no conflict of interests.

References

[1] Ser-Huang Poon, Forecasting volatility in financial markets: a review, Journal of Economic Literature. 41(2)(2003) 478-539.

[2] Wang, Y. F,Predicting stock price using fuzzy grey prediction system, Expert Systems with Applications. 22(1)(2002) 33-38.

[3] Ramalingam Shanmugam, Introduction to time series and forecasting, Technometrics. 39(4)(2002) 426-426.

[4] Guresen, E., Kayakutlu, G., Daim, T. U.Using artificial neural network models in stock market index prediction. Expert Systems with Applications. 38(8)(2011)10389-10397.

[5] Hsieh, T. J., Hsiao, H. F., Yeh, W. C., Forecasting stock markets using wavelet transforms and recurrent neural networks: an integrated system based on artificial bee colony algorithm, Applied Soft Computing Journal. 11(2) (2011) 2510-2525.

[6] Kim, K. J., Han, I., Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index, Expert Systems with Applications. 19(2)(2000) 125-132.

[7] Wang, J., Wang, J., Forecasting stock market indexes using principle component analysis and stochastic time effective neural networks, Neurocomputing. 156(C)(2015) 68-78.

[8] Pradeepkumar, D., Ravi, V., Forecasting financial time series volatility using particle swarm optimization trained quantile regression neural network, Applied Soft Computing. 58(2017)35-52.

[9] Rumelhart, D. E., Hinton, G. E., Williams, R. J., Learning representations by back-propagating errors, Nature. 323(6088)(1986) 533-536.

[10] Reby, D., Lek, S., Dimopoulos, I., Joachim, J., Lauga, J., Aulagnier, S., Artificial neural networks as a classification method in the behavioural sciences, Behavioural Processes. 40(1)(1997) 35-43.

[11] Khashei, M., Bijari, M., Fuzzy artificial neural network p, d, q model for incomplete financial time series forecasting, Journal of Intelligent & Fuzzy Systems Applications in Engineering & Technology. 26(2)(2014) 831-845.

[12] Adhikari, R., Agrawal, R. K., A combination of artificial neural network and random walk models for financial time series forecasting, Neural Computing & Applications. 24(6)(2014) 1441-1449.

[13] Kuremoto, T., Kimura, S., Kobayashi, K., Obayashi, M., Time series forecasting using a deep belief network with restricted boltzmann machines, Neurocomputing. 137(15)(2014) 47-56.

[14] Wang, J., Wang, J., Forecasting stochastic neural network based on financial empirical mode decomposition, Neural Networks. 90(2017) 8-20.

[15] Francis E.H Tay, Lijuan Cao, Application of support vector machines in financial time series forecasting, Journal of University of Electronic Science and Technology of China. 29(4)(2007) 309-317.

[16] N.I. Sapankevych, Ravi Sankar, Time series prediciton using support vector machines: a survey. IEEE Computational Intelligence Magazine. 4(2)(2009) 24-38.

[17] Kazem, A., Sharifi, E., Hussain, F. K., Saberi, M., Hussain, O. K. Support vector regression with chaos-based firefly algorithm for stock market price forecasting. Applied Soft Computing Journal. 13(2)(2013) 947-958.

[18] Lin, T., Horne, B. G., Giles, C. L., How embedded memory in recurrent neural network architectures helps learning long-term temporal dependencies, Neural Networks. 11(5)(1998)861-868.

[19] Chen, P. A., Chang, L. C., Chang, F. J., Reinforced recurrent neural networks for multi-step-ahead flood forecasts, Journal of Hydrology. 497(2013) 71-79.

[20] Guo, T., Xu, Z., Yao, X., Chen, H., Aberer, K., Funaya, K., Robust Online Time Series Prediction with Recurrent Neural Networks, IEEE International Conference on Data Science and Advanced Analytics. IEEE, 9(2016)816-825.

[21] Sundermeyer, M., Schlüter, R., Ney, H., LSTM Neural Networks for Language Modeling, Interspeech. (2012)601-608.

[22] Sundermeyer, M., Ney, H., Schlüter, R., From feedforward to recurrent lstm neural networks for language modeling, IEEE/ACM Transactions on Audio Speech & Language Processing. 23(3)(2015) 517-529.

[23] Huang, N. E., Shen, Z., Long, S. R., Wu, M. C., Shih, H. H., Zheng, Q., et al., The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis, Proceedings Mathematical Physical & Engineering Sciences. 454(1971)(1998) 903-995.

[24] ZHAOHUA WU, NORDEN E. HUANG. Ensemble empirical mode decomposition: a noise-assisted data analysis method. Advances in Adaptive Data Analysis. 1(01)(2005) 1-41.

[25] Torres, M. E., Colominas, M. A., Schlotthauer, G., & Flandrin, P. A complete ensemble empirical mode decomposition with adaptive noise. IEEE International Conference on Acoustics, Speech and Signal Processing. 125(2011)4144-4147.

[26] Bengio, Y., Simard, P., Frasconi, P., Learning long-term dependencies with gradient descent is difficult, IEEE Transactions on Neural Networks. 5(2)(1994) 157.

[27] Sepp Hochreiter, Jürgen Schmidhuber, Long short-term memory, Neural Computation. 9(8)(1997) 1735-1780.

[28] Graves, A., Generating sequences with recurrent neural networks, arXiv preprint. (2013)arXiv:1308.0850.

[29] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R., Dropout: a simple way to prevent neural networks from overfitting, Journal of Machine Learning Research. 15(1)(2014) 1929-1958.

[30] Yarin Gal., A theoretically grounded application of dropout in recurrent neural networks, Statistics. (2015)285-290.

[31] Kingma, D. P., Ba, J., Adam: a method for stochastic optimization, arXivpreprint.(2014)arXiv:1412.6980.

[32] Makridakis, S., Accuracy measures: theoretical and practical concerns ☆, International Journal of Forecasting. 9(4)(1993) 527-529.

[33] Wang, J., Wang, J., Fang, W., Niu, H., Financial time series prediction using elman recurrent random neural networks, Computational Intelligence and Neuroscience. 2016,(2016-5-18), 2016(12) 1-14.