sustainability

MDPI

Article

Reducing Exchange Rate Risks in International Trade: A Hybrid Forecasting Approach of CEEMDAN and Multilayer LSTM

Hualing Lin *, Qiubi Sun and Sheng-Qun Chen

The School of Economics and Management, Fuzhou University, Fuzhou 350108, China; sunsqb@fzu.edu.cn (Q.S.); 1698030554681@fjjxu.edu.cn (S.-Q.C.)

* Correspondence: m16071006@fzu.edu.cn

Received: 4 February 2020; Accepted: 15 March 2020; Published: 20 March 2020

Abstract: In international trade, it is common practice for multinational companies to use financial market instruments, such as financial derivatives and foreign currency debt, to hedge exchange rate risks. Making accurate predictions and decisions on the direction and magnitude of exchange rate movements is a more direct way to reduce exchange rate risks. However, the traditional time series model has many limitations in forecasting exchange rate, which is nonlinear and nonstationary. In this paper, we propose a new hybrid model of complete ensemble empirical mode decomposition (CEEMDAN) based multilayer long short-term memory (MLSTM) networks. It overcomes the shortcomings of the classic methods. CEEMDAN not only solves the mode mixing problem of empirical mode decomposition (EMD), but also solves the residue noise problem which is included in the reconstructed data of ensemble empirical mode decomposition (EEMD) with less computation cost. MLSTM can learning more complex dependences from exchange rate data than the classic model of time series. A lot of experiments have been conducted to measure the performance of the proposed approach among the exchange rates of British pound, the Australian dollar, and the US dollar. In order to get an objective evaluation, we compared the proposed method with several standard approaches or other hybrid models. The experimental results show that the CEEMDAN-based MLSTM (CEEMDAN–MLSTM) goes on better than some state-of-the-art models in terms of several evaluations.

Keywords: exchange rate risks; hybrid forecasting approach; complete ensemble empirical mode decomposition with adaptive noise (CEEMDAN); multilayer long short-term memory (MLSTM)

1. Introduction

Exchange rate risks are one of the main risks in international trade. It can be defined as losses that are caused by the fluctuation of the local currency relative to the currency of the trading nation. After the collapse of the Bretton Woods system, most governments adopted more flexible exchange rate regimes, which are characterized by unanticipated exchange rate fluctuations [1]. Exchange rate movements are affected by many complex factors, including political and economic events. The risks of international trade become more accentuated at the same time as the increase of the exchange rates volatility. The exchange rate risks faced by multinational trading firms include transaction risk, translation risk, and economic risk [2]. For example, related to transaction risk, fluctuations in the exchange rate of the denominated currency caused losses to import and export contracts related to accounts receivable. Also, the risks attract the attention of national policy-makers who need to embrace the challenge related to the exchange rates volatility. To reduce these risks and avoid the returns from overseas trade being eroded, the use of financial market instruments, such as financial derivatives (forward, futures, options, etc.) and foreign currency debt, is the most important way for

multinational trading firms to hedge exchange rate risks [3]. The premise of using these financial market instruments for hedging risk is that the correct measurement of risk has been proven to be difficult by Van Deventer et al. [4]. Making accurate forecasts and decisions of the direction and magnitude of fluctuations is another more direct and efficient approach for reducing exchange rate risks. This research will propose a new approach to improve the accuracy of short-term exchange rate forecasting for reducing exchange rate risks.

Over the course of the past 40 years, many classical time series analysis models such as AR, ARMA, ARCH, and GARCH have been widely used in financial time series analysis [5–7]. The linear autoregressive (AR) model provides the flexibility to model stationary time series. The autoregressive moving average (ARMA) model assumes a linear relationship between the lag variables [8]. It has poor prediction accuracy for nonlinear and nonstationary time series. Especially when time trends and seasonal features appear in nonstationary time series, the predictive performance of the ARMA model degrades significantly. Box and Jenkins proposed the famous autoregressive integrated moving average (ARIMA) model in the 1970s [9]. This model eliminates or reduces the first-order nonstationarity by differencing. However, the differencing usually amplifies the high frequency noise in the time series, so much effort is required to determine the order of the ARIMA model. Engle introduced the autoregressive conditional heteroskedasticity (ARCH) model to capture second-order nonstationarity [10]. The generalized autoregressive conditional heteroskedasticity (GARCH) model is an extension of ARCH that represents the variance of the error term as a function of its autoregressive terms [11]. However, all these methods tend to be limited for nonlinear and nonstationary time series forecasting.

In the last twenty years, researchers in machine learning have tried to solve the problems of time series forecasting with machine learning methods and achieved certain results. These methods include random forests, support vector machines, Bayesian, and the like. Lin et al. proposed a random forests-based extreme learning machine ensemble model in multiregime time series forecasting [12]. Cao proposed using the support vector machines (SVMs) experts for time series forecasting [13]. Balabin and Lomakina proposed a model based on SVM-based generalized regression for time series forecasting, such as support vector regression (SVR) and least-squares support vector machines (LSSVMs). A novel forecasting approach with a Bayesian-regularized artificial neural networks (BR-ANN) was proposed by Yan et al. [14]. The methods mentioned above play an important role in time series forecasting which have small data size and smooth features. However, the results of predictions are not satisfactory when the data is large, nonstationary, or nonlinear [15].

In recent years, the research of artificial neural networks has made breakthrough progress. Deep learning can deal with many practical problems that are difficult to solved by conventional methods. It has been widely used in financial time series forecasting [16–20]. The neural network models are known to be effective in stock market dynamic predictions [21,22]. Studies have shown that the models of deep learning can approximate nonlinear functions with arbitrary precision while there are enough hidden nodes (neurons). The recurrent neural network (RNN), one kind of the deep learning with time series processing capabilities, has naturally become a strong contender for these methods of classic approaches of exchange rate forecasting [23–25].

In the classic RNN, the information of each time step is stored in the internal state of the network. When training, it needs to learn in the opposite direction of time. The multiplication of the derivative of the activation function will lead to the disappearance of the gradient or the phenomenon of gradient explosion. The long short-term memory (LSTM) proposed by Hochreiter and Schmidhuber as an extension of RNN can learn the long-term dependence of data, and is widely used for speech recognition, natural language processing, image identification, time series forecasting, and other fields [26–29]. Dai et al. proposed trajectory prediction modeling of vehicle interactions through an improved LSTM model [30]. Hao et al. used an improved LSTM model for pedestrian trajectory prediction [31], but their performance on financial time series forecasting issues is not satisfactory. This

single-layered architecture does not effectively demonstrate the complex features of exchange rate data, especially when trying to deal with nonstationary, nonlinear and long-interval time series data [32,33].

Empirical mode decomposition (EMD) is an adaptive signal decomposition algorithm for nonlinear, nonstationary signals [34]. EMD cannot effectively decompose the nonstationary signals if there are not enough extreme points. In order to solve the mode mixing problem of EMD, Wu and Huang proposed a white-noise-assisted data analysis method called ensemble empirical mode decomposition (EEMD) [35]. Compared to EMD, EEMD eliminates the influences of mode mixing, but still retains some noise in the intrinsic mode functions (IMFs) [36]. The complete ensemble empirical mode decomposition with adaptive noise (CEEMDAN) algorithm obtains a set of noise-containing signals by adding multiple sets of independent and identically distributed adaptive white noise with finite variance constraints on the original signal, and then decomposes with EMD [37]. It can further reduce the number of iterations, compress the frequency aliasing region, and improve the convergence performance. It has better performance than EEMD on decomposition of the nonstationary signal.

Exchange rate forecasting is based on current and past information in the foreign exchange market to predict future exchange rate behavior. As a financial time series, the exchange rate data owns not only the characteristics of general time series such as autocorrelation, trend, seasonality, and random noise, but also features of financial time series such as nonlinearity, nonstationary, and volatility clustering [38]. Meese and Rogoff compared the out-of-sample forecasting accuracy of exchange rate prediction methods. They found that a random walk model performs as well as other models at exchange rate forecast within 12 months [39]. Galeshchuk further expanded the set of models to include Taylor rule fundamentals, yield curve factors, and incorporate shadow rates and risk and liquidity factors [40]. Wright proposed Bayesian Model Averaging and found that the prediction error is smaller than the random walk model [41].

The goal of complete ensemble empirical mode decomposition with adaptive noise based on multilayer long short-term memory (CEEMDAN–MLSTM) proposed in this paper is to achieve better performance on the exchange rate forecasts. The complete ensemble empirical mode decomposition with CEEMDAN is combined with multilayer long short-term memory (MLSTM) to predict the exchange rate. A brief description of its work process is as follows. First, the positive and negative white noise pairs with the same amplitude, and opposite phase are added to the original signal to reduce the non-stationarity of exchange rate data. Then, the data is decomposed through EMD to produce a series of intrinsic mode functions (IMFs) with different characteristic scales. These IMFs represent exchange rate fluctuations caused by different investors at different times. Finally, we input the IMFs data into the LSTM network model and output the results. In order to achieve an objective evaluation, we compared the performance of CEEMDAN–MLSTM with other reference models using the same dataset and the same experimental conditions through different error measurement. The results show that CEEMDAN–MLSTM is more suitable for the analysis of nonlinear nonstationary signals because it signally reduces the number of iterations and increases the reconstruction accuracy. Details on the proposed method are discussed in later sections.

The remainder of this paper is organized as follows. In Section 2, we introduce the EMD, EEMD, CEEMDAN signal decomposition mechanism and the architecture of the recurrent neural network. Next, we demonstrate how CEEMDAN–MLSTM works for exchange rate forecasting. In Section 3, we carried out a series of experiments, including data preprocessing, data decomposition, evaluation of indicators, and description and analysis of experimental results. In Section 4, the last section, the conclusion can be seen.

2. Methodology

2.1. EMD, EEMD, and CEEMDAN

Empirical mode decomposition (EMD) was proposed by Huang et al. in 1998 [34]. Its core function is the smooth processing of time signals, and the time–time spectrum map it obtains by performing a

Hilbert transformation for a nonstationary time series, breaking complex signals into limited IMFs [42]. The decomposed IMFs contain the characteristic of original signals for different time scales. Any time signal can be broken down into the sum of several IMF components.

The IMF has two constraints:

1. The number of extrema and the number of zero-crossings of IMF must be equal or not more than one difference.
2. The mean of the upper envelope defined by the local maxima and the lower envelope formed by the local minima is zero.

The sifting process of EMD is as follows:

1. Find out all the maxima and minima of the original signal A, and fit the upper, lower envelopes and a mean line *M(t)* of the original data with three spline interpolation functions. The distribution of the three lines is shown in Figure 1.

$$M(t) = U(t) - L(t)$$

where $U(t)$ and $L(t)$ are the upper envelope and the lower envelope, respectively.

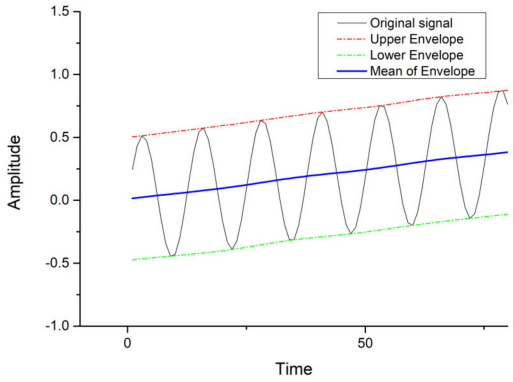


Figure 1. The diagram of the envelope: the original data in the thin black line, the upper and the lower envelopes in the dot-dashed lines, and their mean in the blue solid line.

2. Subtract the original data A from the mean *M(t)* of the envelopes. If the new data meets the IMF constraints, the data is the *IMF*.

$$IMF(t) = X(t) - M(t)$$

3. The result of subtracting the original data from the IMF is the residue data. Using the residue data as the new data and repeating the above process, the next *IMF* can be obtained.

$$R_K(t) = X(t) - \sum_{i=1}^{K} IMF_i$$

$$R_{K+1}(t) = R_K(t) - IMF_{K+1}$$

where $R_K(t)$ is the residue after K-th decomposition.

One of the drawbacks of EMD is mode mixing, which means that EMD cannot untangle two or more frequency components. When mode mixing occurs, the single IMF consists of signals of disparate scales, or a similar scale signal residing in different IMFs. In order to alleviate this drawback, in 2009,

Huang et al. proposed an improved algorithm called ensemble empirical mode decomposition (EEMD), which is a noise-assisted data analysis method [35]. After a sufficient number of trials, the averaged-out white noise is added to the original signal. This approach utilizes the statistical characteristics of white noise and can separate scales naturally without any a priori subjective criterion selection. By changing the distribution of the signal extrema, we can obtain the upper and lower envelopes corresponding to the signal characteristics. The process of EEMD is as follows:

1. Add the Gaussian white noise to the original data.
2. The decomposition of EMD is performed to obtain the IMF components.
3. Repeat steps 1 and 2, adding a new normal distribution white noise data each time.
4. Integrating the obtained IMFs according to the order to get the mean, each component of EEMD can be obtained.

EEMD overcomes the problem of mode mixing, but, the reconstructed data includes residual noise still, and different realizations of signal plus noise may produce different numbers of modes. To overcome drawbacks of EEMD, Torres et al. proposed a new approach of complete ensemble empirical mode decomposition with adaptive noise (CEEMD) for signal decomposition [36]. During CEEMDAN, the original data is added to a special type of white noise that includes pairs of positive and negative noises, and then engages in EMD processing. CEEMDAN uses a strategy called "divide and rule" to transform the complex problems into one of the relatively simple components of independent prediction. Compared with EMD and EEMD, this method effectively largely eliminated the problem of mode mixing, and no matter how many times the decomposition, the reconstruction error of the signal is almost zero. CEEMDAN is not only better in completeness, but also solves the problem of low decomposition efficiency, greatly reducing the calculation cost [40]. The algorithm is described as follows:

1. Generate a particular white noise and add it to original signal: $[x(t) + w_0]$ Here $x(t)$ represent the original signal and w_0 is the first white noise of finite variance. Then, we extract all the IMFs from the new series by EMD. The mean of these IMFs is the first mode of CEEMDAN.

$$\overline{imf_1} = \frac{1}{n}\sum_{j=1}^{n} imf_1^j(t)$$

   here, n represent the number of modes, while $imf_1^j(t)$(j=1,2, … … n) indicates the whole modes decomposed by EMD. $\overline{imf_1}$ is the first mode of CEEMDAN and equal to the mean of $imf_1^j(t)$.

2. Calculate the first-order residue:

$$r_1(t) = x(t) - \overline{imf}_1$$

3. Decompose the residue to get the second-order IMF and calculate the second-order residue:

$$\overline{imf_2} = \frac{1}{n}\sum_{j=1}^{n} E_1\left\{r_1(t) + \varepsilon_1 E_1\left[w_2^j(t)\right]\right\} r_2(t) = r_1(t) - \overline{imf_2}$$

4. Decompose the k-th residue and extract the first mode, and the k-th mode can be obtained using the following equation (k = 2, 3, … K):

$$\overline{imf_k(t)} = \frac{1}{n}\sum_{j=1}^{n} E_1\left(r_{k-1}(t) + w_{k-1}^j E_{k-1}\left(\varepsilon^j(t)\right)\right)$$

While $k = 2, \ldots, K$, calculate the k-th residue:

$$r_k(t) = x(t) - \sum_{i=1}^{k} imf_i$$

where k is the number of IMFs. On different time-scales, all of the IMFs constitute the characteristics of the original signal. The residue clearly shows the trend of the original signal, which is smoother than the original data, and effectively reduces the prediction error.

2.2. RNN, LSTM, MRNN and MLSTM

2.2.1. RNN

Recurrent neural networks (RNN) are widely used in time data modeling. Multilayer perceptron (MLP) or other feedforward neural networks can only map inputs to output vectors, while RNN can map the entire history of previously entered data to each output. The output of the network at time t is not only related to the input of time t, but also to the recursive signal before time t. During backpropagation, the RNN meets a challenge of gradient disappearance and explosion. Although, RNNs are established to learn the dependencies of time series. Empirical evidence suggests that RNNs cannot learn long-term dependencies because of the problem of gradient, but only could process short-term time series. The architecture of RNNs is illustrated in Figure 2.

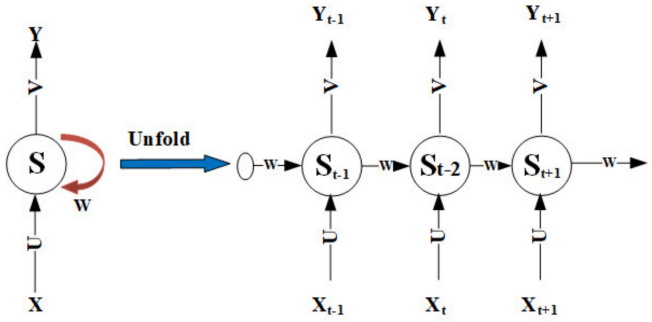


Figure 2. Structure of recurrent neural network (RNN); X is current input; Y is the output of neural network; U, V, and W are parameters of the neural network; S is the neuron state of the hidden layer, which stores the memory of the neural network; The state S at time t is related to the current input X and the memory at time $t-1$. The formula is expressed as follows: The state S at time t is related to the current input X and the memory at time $t-1$. The formula is expressed as:$S_t = f(UX_{t-1} + WS_{t-1})$

2.2.2. LSTM

Long short-term memory (LSTM) is an elegant variant of RNN architecture and used to solve the problems of gradient disappearance of RNN. The LSTM controls the addition or deletion of state information of cells through a module that includes the gating mechanism. The cell state is equivalent to the path of information transmission and run directly on the entire chain. The gate structure selectively allows the information to pass. Figure 3 shows the architecture of a single LSTM block.

Calculate the cell state and hidden state of the t-th time step as follows:

The first step in our LSTM is to decide what information we are going to throw away from the cell state. This decision is made by a sigmoid layer called the "forget gate layer." It looks at h_{t-1} and x_t , and outputs a number between 0 and 1 for each number in the cell state C_{t-1}. 1 represents "completely keep this", while 0 represents "completely get rid of this."

$$f_t = \sigma\left(X_t U^f + h_{t-1} W^f + b_f\right)$$

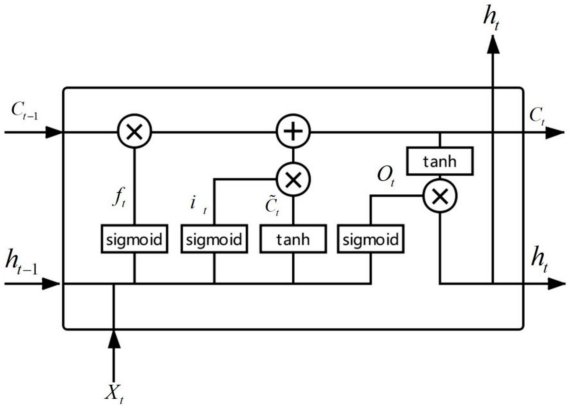


Figure 3. Long short-term memory (LSTM) network structure.

The next step is to decide what new information we are going to store in the cell state. This has two parts. First, a sigmoid layer called the "input gate layer" decides which values we will update. Next, a tanh layer creates a vector of new candidate values; $\widetilde{C}_t$ can be added to the cell state. Then, update the cell status from C_{t-1} to C_t.

$$i_t = \sigma\left(X_t U^i + h_{t-1} W^i + b_i\right)$$

$$\widetilde{C}_t = \tanh(X_t U^c + h_{t-1} W^c + b_c)$$

$$C_t = C_{t-1} \times f_t + i_t \times \widetilde{C}_t$$

Finally, the output gate is used to determine the value of the next hidden state, which contains the previously entered information. First, we pass the previous hidden state and current input to the sigmoid function, and then pass the newly obtained cell state to the tanh function. Finally, the output of tanh is multiplied by the output of sigmoid to determine the information that should be carried in the hidden state. The hidden state is then used as the output of the current cell, passing the new cell state and the new hidden state to the next LSTM cell.

$$O_t = \sigma\left(X_t U^0 + S_{t-1} W^0 + b_0\right)$$

$$h_t = O_t \times \tanh(C_t)$$

2.2.3. CEEMDAN–MLSTM

Multilayer RNNs increase the number of hidden layers. The parameters of each layer of the network can be shared. Compared with the single-layer neural network, the multilayer neural network has a more powerful generalization capability and can solve the more complex problems.

The goal of the proposed method is to improve the accuracy and performance of exchange rate forecasting. The essence of this approach is a multilayer LSTM network in which multiple groups of LSTM units are stacked. In multilayer LSTM, the output of the previous layer is the input of the next layer. The proposed model embeds a dense layer before the output layer. The dense layer is a fully connected layer. It maps the neurons in one cell to the next cells, which are both in the next cell of the same layer and the next layer. The output tensor of the last LSTM unit is extracted from the layer before dense. The tensor is used as the input data of the dense layer and then linearly transformed to output the prediction of the exchange rate. The activation function for most fully connected units is the function of Relu. The architecture of multilayer LSTM is illustrated in Figure 4.

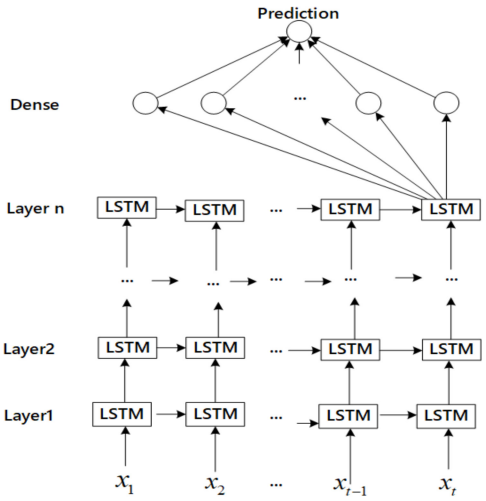


Figure 4. Framework and process diagram of multilayer LSTM model.

To implement the exchange rates forecasting by the proposed method, the steps are as follows:

1. Decompose the original data into relatively simple IMFs by CEEMDAN.
2. Input the IMFs into the multilayer LSTM neural network separately and predict each extracted IMF.
3. Finally, sum up the IMF and residual of each training. Then, we get the expected prediction.

$$prediction(t) = \sum_{i=1}^{k} \overline{imf_k} + r_k, (k = 1, 2 \cdots\cdots l)$$

The model framework and business process diagram are described in Figure 5.

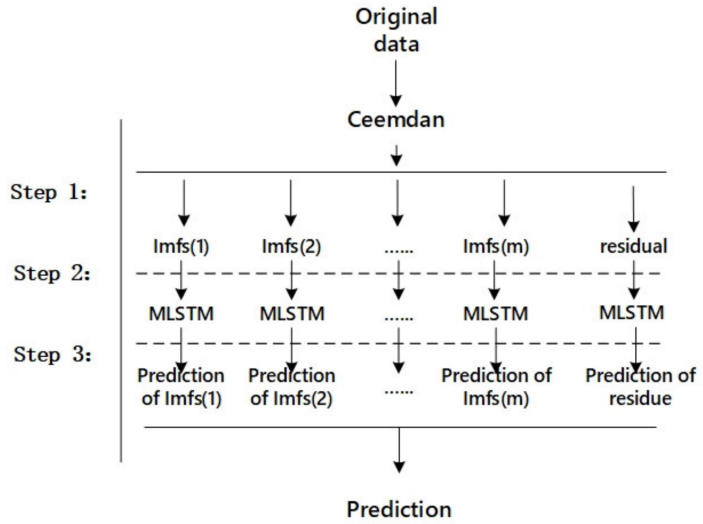


Figure 5. The architecture of complete ensemble empirical mode decomposition based multilayer LSTM(CEEMDAN–MLSTM).

3. Experiment

To make the evaluation more objective and accurate, we compared the CEEDMAN–MLSTM with other prediction models through a series of experiments on real data sets. We analyzed the effects of deep learning models under different training strategies, network architecture, and network depths. Figure 6 shows the flowchart of multilayer LSTM network training.

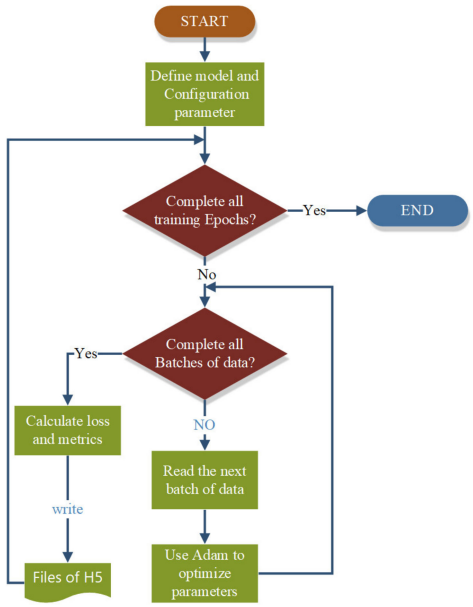


Figure 6. The flow chart of the neural network training.

3.1. Data Preparing and Description

According to previous analysis, we should verify the effectiveness and stability of the proposed approach for the real datasets. Considering two characteristics of convertibility and influence, we chose the real exchange rate data of GBP–USD and USD–AUD from 4 January 1971 to 25 August 2017 for empirical research on exchange rate forecasting. In the subsequent experiments, the trends of exchange rate were analyzed and forecasted. Figure 7 shows the trend and fluctuation amplitude of the USD–AUD exchange rate.

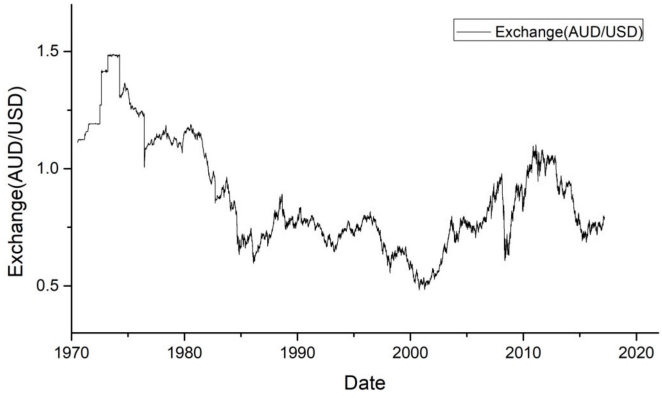


Figure 7. The trend and fluctuation amplitude of the USD–AUD exchange rate.

Data preprocessing is an important step in machine learning tasks. It includes various functions such as filter or extract, scale, normalize, or shuffle. Before these experiments, we did the necessary preprocessing of the initial data. First, we did a preliminary exploration of the initial data and observed the distribution. Secondly, we cleaned the missing values in the data and normalized the data. Finally, we divided the cleaned data into training sets and test sets.

These experiments adopted the n-step-ahead prediction strategy for all models. It made predictions over the latest k days of exchange rate available, always forecasting just one day ahead. After preprocessing, the data set is converted to $[(N - k) \times (k + 1)]$ (N: dataset length, k: steps) Matrix. Any row of the matrix represents one of the training samples. The next k of observations for each sample were used as input to the model, and the first data is validation data. In order to evaluate the

efficiency of different models in exchange rate forecasting, we divided the data into two parts: training set and test set. In the following experiments, the proportion of training set is 80% of total data, and the remaining 20% belong to the test set.

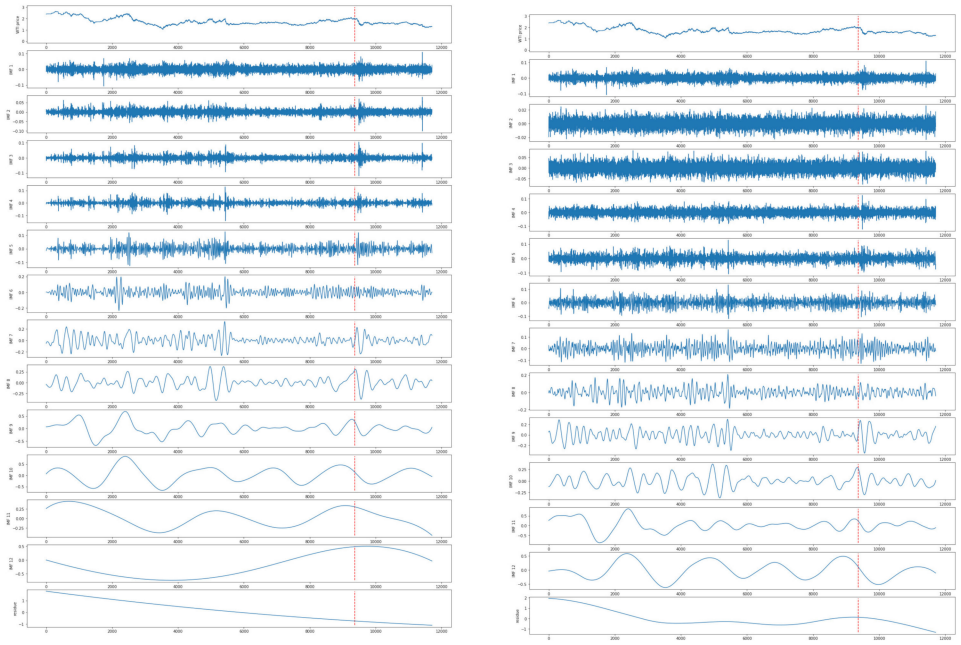
3.2. Data Decomposition

In order to evaluate the influence of different decomposition methods on prediction, the data sets were decomposed by the EMD, EEMD, and CEEMDAN methods respectively, and three different IMFs were obtained. To investigate the impact of the parameters of CEEMDAN, some experiments were conducted on exchange rate data with one-step-ahead forecasting. We ran the proposed hybrid model with the parameters of ensemble_size in the range of [4 200, 300, 500, 1000, 2000] and noise_strength in the range of [0.01, 0.02, 0.03, 0.05, 0.10]. By synthesizing the metrics, it was finally determined that the parameters of noise strength = 0.05 and ensemble_size = 200. This means that the standard deviation of 0.05 and quantity of 100 white noise data will be added to the exchange rate data. Other parameters have little influence on the model, so we directly adopted the default settings. The parameters description and settings of EMD, EEMD, and CEEMDAN are shown in Tables 1 and 2.

Table 1. Decomposition Parameters Description.

Parameter	Description
spline_kind	Defines type of spline, which connects extrema
nbsym	Number of extrema used in boundary mirroring
max_imf	IMF number to which decomposition should be performed
ensemble_size	Number of trials or EMD performance with added noise
noise_strength	Standard deviation of the Gaussian random numbers used as additional noise.

The time–frequency spectrums of IMFs obtained by EEMD, CEEMDAN are shown in Figure 8.



(a) (b)

Figure 8. The time–frequency spectrums of IMFs obtained by ensemble empirical mode decomposition(EEMD) and Complete EEMD with adaptive noise(CEEMDAN); (**a**) the IMFs and residue of the USD–GBP exchange rate obtained by EEMD; (**b**) the IMFs and residue of the USD–GBP exchange rate obtained by CEEMDAN.

Table 2. Parameter Setting of Decomposition.

Method	nbsym	max_imf	trials	noise_width/epsilon
EMD	2	ALL	–	–
EEMD	2	ALL	100	0.05
CEEMDAN	2	ALL	100	0.05

Note: EMD: empirical mode decomposition; EEMD: ensemble EMD; CEEMDAN: Complete EEMD with adaptive noise.

3.3. Building of MLSTM

The end of the data preprocessing and decomposition means that the data preparation has been completed. Once the model is built, we can feed the data to the model for training. In Section 2, we have researched the structure and workflow of MLSTM. In this subsection, we will discuss the initialization of several key parameters that directly affect the efficiency of our proposed deep learning model, and briefly illustrate the dependencies of components in MLSTM.

3.3.1. Loss

As with machine learning, deep learning also learns the parameters of complex problems through the loss. In the field of machine learning, with the help of some functions of optimization, the loss can well measure the gap between the model predictions and the actual observations. The essence of training of neural networks is the learned process of weight parameters. The training of neural networks makes the functions of loss gradually converge, and the model is continuously optimized and the prediction error becomes smaller and smaller. The neural network returns the error from the loss through the backpropagation algorithm, and then optimizes the hyperparameters in the network by means like the gradient descent. The experiments in this paper are to evaluate the prediction accuracy of the methods, which is a type of question about regression, so we chose MSE as the loss function.

3.3.2. Learning Rate

Learning rate (LR) refers to the magnitude of the update network weight in the optimization algorithm. The learning rate can be constant, gradually decreasing, momentum-based, or adaptive. Different optimization algorithms determine different learning rates. When the learning rate is too large, the model may not converge, and the loss will continue to fluctuate up and down. If the learning rate is too small, the model will converge slowly and require longer training. These experiments use the optimization algorithm of Adam, which can design an independent adaptive learning rate for different parameters by calculating first-order moment estimation and second-order moment estimation of gradient. For comparison purposes, we set the learning rate of the deep learning models in the experiment to 0.01.

3.3.3. Batch_size

Batch_size is the number of samples that are sent to the neural network model for each training. In neural networks, the larger Batch_size usually makes the network converge faster. However, setting the oversized Batch_size may result in insufficient memory or a program kernel crash. In the experiment, we set the parameters of epochs and batch_size to 100 and 256, respectively.

3.3.4. Optimizer

Since the adaptive optimization algorithm can quickly find the correct target direction in the parameter update, the loss function can converge quickly. The standard stochastic gradient descent (SGD), nesterov accelerated gradient(NAG) and momentum methods are difficult to find the correct direction, so the convergence is slow. Experience has shown that adaptive moment estimation (Adam) has faster convergence and more effective learning effects than the other adaptive learning

rate algorithms. Adam solves the problems in other optimization techniques, such as learning rate disappearing, convergence is too slow, and the parameter update of the high variance causes the function of loss to fluctuate greatly. Considering that the convergence of the CEEDMAN–MLSTM model is more complicated than the common neural network, we chose Adam as the optimizer.

3.3.5. Metrics

We chose mean absolute error (MAE), root mean squared error (RMSE) and mean absolute error percentage (MAPE) as the performance metrics to measure a model"s relevance:

1. MAE:

$$MAE(y, \hat{y}) = \frac{\sum_{i=1}^{N} \left| y_i - \hat{y}_i \right|}{N}$$

MAE can better reflect the actual situation of the error of prediction.

2. RMSE:

$$RMSE(y, \hat{y}) = \sqrt{\frac{\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}{N}}$$

RMSE is used to measure the deviation between observations and real values and is the most commonly used indicator of forecasts by machine learning models.

3. MAPE:

$$MAPE(y, \hat{y}) = \frac{1}{N} \sum_{i=1}^{N} \left| \frac{y_i - \hat{y}_i}{y_i} \right|$$

MAPE considers not only the absolute error between the predicted and the real value, but also the relative distance between the error and the true value. The letter y in the above formula represents the true value, $\hat{y}$ represents the predicted value, and N is the number of samples.

3.3.6. Dependencies in MLSTM

We trace back the dependencies between the internal components of the MLSTM based on the visualization obtained after model training. Figure 9 is the computation graph containing many nodes and connecting lines. These nodes represent the computations, while the directional lines represent some data flows. Obviously, this MLSTM is a multilayer deep learning model consisting of two LSTM hidden layers, a Dense layer, an input layer and an output layer. The bottom oval section represents the input node. The training data is provided in the form of a tensor to the first hidden layer of the MLSTM through the input nodes. After computation, the output of first hidden layer is converted to the input of the second hidden layer. Finishing calculations of all hidden layers , the dataflow is sent to the dense layer, where the dataflow is compressed into an intermediate prediction and input to the loss node for loss calculation. The orange lines represent the backpropagation of the data flows. Adam is responsible for optimizing the computation of parameters tuning. The updating information of parameters flows from the output layer to the input layer. It is a back propagation process. What has been discussed above is an iteration of MLSTM training.

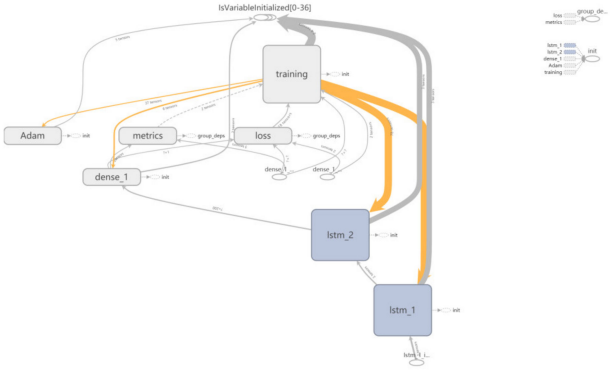


Figure 9. The computation graph of the CEEMDAN–MLSTM model.

3.4. Experimental Results and Analysis

We conducted two groups of experiments. In the first group of experiments, the undecomposed data of exchange rates were imported into the models such as ARIMA, Bayesian, SVM, RNN, LSTM, and Multilayer LSTM. In the second group of experiments, we input the decomposed IMFs into the RNN, MRNN, LSTM, and MLSTM for training and evaluation.

3.4.1. Prediction and Analysis Based on Undecomposed Data

We input the cleaned but undecomposed data into several classic models and the proposed model, such as ARIMA, RNN, MRNN (Multilayer RNN), LSTM, and MLSTM (Multilayer LSTM), respectively. Will different models result in a different forecast? Both Figure 10 and Table 3 illustrate the answers. Table 3 shows us that, among the seven methods, the famous ARIMA model has the lowest predictive performance score. It indicates that ARIMA is not suitable for forecasting exchange rate data that is nonlinear and nonstationary. Machine learning algorithms such as support vector machines (SVM), Bayesian, and artificial neural networks (ANN) perform better than ARIMA models. However, they do not perform as well as several deep learning models. The LSTM model has better performance than the classic RNN, even if both of them belong to recurrent neural networks. MLSTM achieves the best performance. This phenomenon indicates that multilayer architecture of LSTM performs better than the single-layer's. In other words, the multilayer network has better generalization than single in solving complex problems. Yet, the former needs more parameters to be trained than the latter and requires more learning time. From the training time point of view, the more complex the structure of network, the longer the training takes. LSTM training takes 43% more time than RNN. A network with a two-layer LSTM stack structure consumes approximately 40% more time than a single-layer LSTM in training.

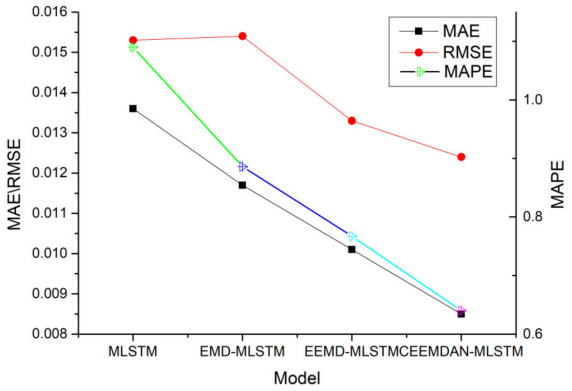


Figure 10. The impact of different decomposition forms of data on exchange rate forecasting.

Table 3. Results of exchange rate forecasting based on undecomposed data.

Model	Number of Hidden Layers	Number of Hidden Units	MAE	RMSE	MAPE(%)
ARIMA	-	-	0.087	0.116	6.148
Bayesian	-	-	0.034	0.038	2.327
SVM	-	-	0.035	0.040	2.629
RNN	1	200	0.032	0.036	2.598
MRNN	2	200,200	0.023	0.025	1.836
LSTM	1	200	0.020	0.021	1.614
MLSTM	2	200,200	0.014	0.015	1.090

Note: ARIMA: autoregressive integrated moving average; SVM: support vector machine; RNN: recurrent neural network; MRNN: multilayer RNN; LSTM: long short-term memory neural network; MLSTM: multilayer LSTM.

3.4.2. Prediction and Analysis Based on Decomposition Data

We conducted another group of experiments, which purpose was to measure the effects of EMD, EEMD, and CEEMDAN methods on improving the accuracy of exchange rate forecasting. The experimental steps were developed as follows:

1. Decompose the three preprocessed identical exchange rate data by EMD, EEMD, and CEEMDAN methods to obtain three sets of different IMFs.
2. Divide each group of IMFs into training set and test set, and input the training set into RNN, MRNN, LSTM, and MLSTM models for training.
3. Enter the test data set and observation values into the trained deep learning model to obtain the final evaluation result.

The performance of different methods in the GBP–USD exchange rate forecasting are shown in Tables 4–6. As can be seen from Table 4, the performance of the EMD-LSTM model is better than that of the EMD-RNN. A deep learning model using a multi-layer architecture is better than a model using a single-layer structure. Comparing the previous predictions of single model, it is easy to observe that the predictions based on EMD are better than the undecomposed data.

Table 4. Prediction based on EMD decomposition data.

Model	Number of Hidden Layers	No. of Hidden Units	MAE	RMSE	MAPE
EMD-RNN	1	200	0.017	0.023	0.01304
EMD-MRNN	2	200,200	0.021	0.025	0.01590
EMD-LSTM	1	200	0.020	0.024	0.01456
EMD-MLSTM	2	200,200	0.012	0.015	0.00886

Note: EMD-RNN: empirical mode decomposition based recurrent neural network; EMD-MRNN: EMD-based multilayer RNN; EMD-LSTM: EMD-based LSTM; EMD-MLSTM: EMD-based LSTM.

Obviously, Tables 5 and 6 have some new information besides reaching similar conclusions as Table 4. The CEEMDAN-based hybrid method performs better than the other two types of hybrid methods in exchange rate prediction tasks. Figure 10 shows the predictive performance of the three types of hybrid methods.

Table 5. Prediction based on EEMD decomposition data.

Model	Number of Hidden Layers	Number of Hidden Units	MAE	RMSE	MAPE
EEMD-RNN	1	200	0.022	0.018	0.0136
EEMD-MRNN	2	200,200	0.013	0.016	0.0095
EEMD-LSTM	1	200	0.013	0.017	0.0101
EEMD-MLSTM	2	200,200	0.010	0.013	0.0077

Note: EEMD-RNN: ensemble EMD based RNN; EEMD-MRNN: EEMD-based multilayer RNN; EEMD-LSTM: EEMD-based LSTM; EEMD-MLSTM: EEMD-based LSTM.

Table 6. Prediction based on CEEMDAN decomposition data.

Model	Number of Hidden Layers	Number of Hidden Units	MAE	RMSE	MAPE
CEEMDAN—RNN	1	200	0.111	0.015	0.0083
CEEMDAN–MRNN	2	200,200	0.012	0.016	0.0088
CEEMDAN–LSTM	1	200	0.014	0.011	0.0091
CEEMDAN–MLSTM	2	200,200	0.009	0.012	0.0064

Note: CEEMDAN-RNN: empirical mode decomposition based recurrent neural network; CEEMDAN-MRNN: CEEMDAN-based multilayer RNN; CEEMDAN-LSTM: CEEMDAN-based LSTM; CEEMDAN-MLSTM: CEEMDAN-based multilayer LSTM.

From the experimental results shown in Tables 4–6 and Figure 10, we can see that models using decomposed data perform better than undecomposed data. It is that the EMD, EEMD, and CEEMD solve the problem of high-volatility, nonstationary exchange rate data, which is difficult to be processed. These methods can decompose the fluctuations and trends of different characteristic scales layer by layer from the original exchange rate data. Then, we integrated each predicted IMF for reconstruction, so a more accurate prediction was obtained. Table 6 tells us that the CEEMDAN combined deep learning model is used for exchange rate forecasting and has better performance than EMD and EEMD.

Comparing the approach of CEEMDAN–MLSTM with the others, it is found that the loss function converges the fastest during training, because CEEMDAN not only solves the problem of exchange rate data due to nonlinear and nonstationary features, but also solves the problem of frequent appearance of mode mixing. CEEMDAN–MLSTM shows better performance than other methods. We tested the AUD–USD data set and reached the same conclusion.

3.5. Discussion

3.5.1. Performance of Different Step-Ahead Forecasts

Here, we discuss the performance of different step-ahead forecasts. In previous experiments, we verified the effectiveness of our proposed approach in one-step-ahead forecasts. However, in international trade, not only one step in advance is required, but even two, three, and multi-steps ahead are required. Does the proposed method still work? We started a new set of experiments to measure the performance of the proposed method on multistep prediction. We set four horizons in this set of experiments. Horizon = 1, 2, 10, 20 denote the one-day-, two-day-, ten-day-, and thirty-day-ahead forecast, respectively. The results are shown in Table 7. From this table, we can see that, among these three metrics, MAE and RMSE are used to measure the absolute error between the predictions and the observations, which is closely related to the data scale. As seen from the previous equations, MAPE is a measure of scale-independency and interpretability. Therefore, these metrics should be analyzed together. Overall, our proposed model still achieves the best of all methods on three metrics. The value of metrics is positively related to the level. This shows that our proposed method is still ahead of other methods in multistep prediction. In the multistep forecast of exchange rate based on the daily data, all metrics maintain low values, which shows that multistep forecast of exchange rate is feasible and meaningful. It is easy to observe that, as the step size increases, all metrics increase and the prediction accuracy decreases. It is interesting that the value of MAPE increases faster than the other two metrics. The former phenomenon indicates that the prediction error increases with the number of steps, due to the loss of information and lack of intermediate daily data. The latter is caused by the change in the data. In the observation period, the exchange rate of the British pound to the US dollar showed a downward trend. When the observations go down, MAPE naturally increases.

Table 7. Performance of different step-ahead forecasts.

MAE Horizon	**Methods**			
	CEEMDAN–MLSTM	**CEEMDAN–LSTM**	**CEEMDAN–MRNN**	**CEEMDAN–RNN**
One	0.0096	0.0145	0.0124	0.1116
Two	0.0124	0.0153	0.0161	0.0252
Three	0.0203	0.0232	0.0306	0.0317
Four	0.0406	0.0543	0.0589	0.0411
RMSE Horizon	CEEMDAN–MLSTM	CEEMDAN–LSTM	CEEMDAN–MRNN	CEEMDAN–RNN
One	0.0122	0.0157	0.0168	0.0157
Two	0.0143	0.0183	0.0206	0.0241
Three	0.0189	0.0259	0.0317	0.0345
Four	0.0559	0.0625	0.0646	0.0759
MAPE Horizon	CEEMDAN–MLSTM	CEEMDAN–LSTM	CEEMDAN–MRNN	CEEMDAN–RNN
One	0.0064	0.0091	0.0088	0.0083
Two	0.0109	0.0118	0.0133	0.0147
Three	0.0347	0.0412	0.0462	0.0512
Four	0.1056	0.1263	0.1502	0.1753

3.5.2. Impact of Different Lag Orders

In preceding experiments, we set the proposed model to a fixed number of lag order of 38 by trial and error. Here, we will discuss the impact of different lag orders to the performance of CEEMDAN–MLSTM. We input the exchange rate data organized into different lag orders into CEEMDAN–MLSTM and obtained different outputs, including the metrics mentioned above. All the metrics were used to evaluate the impact of different numbers of lag orders on model performance. We use a segmented strategy to implement our plan. The experiment was conducted in two stages. A total of thirteen experiments were completed in the first stage, and the lag order of these experiments is based on the thirteen numbers of {1, 4, 8, 12, 16, 20, 24, 28, 32, 36, 40, 44, 48}. After completing the first phase of the experiment, a synthetical and optimal solution (lag order = 36) was obtained. In the second phase, we limited the range of lag order from 33 to 39. In this stage, we completed a total of 7 experiments and obtained the optimal solution (lag order = 38). The result is shown in Figure 11.

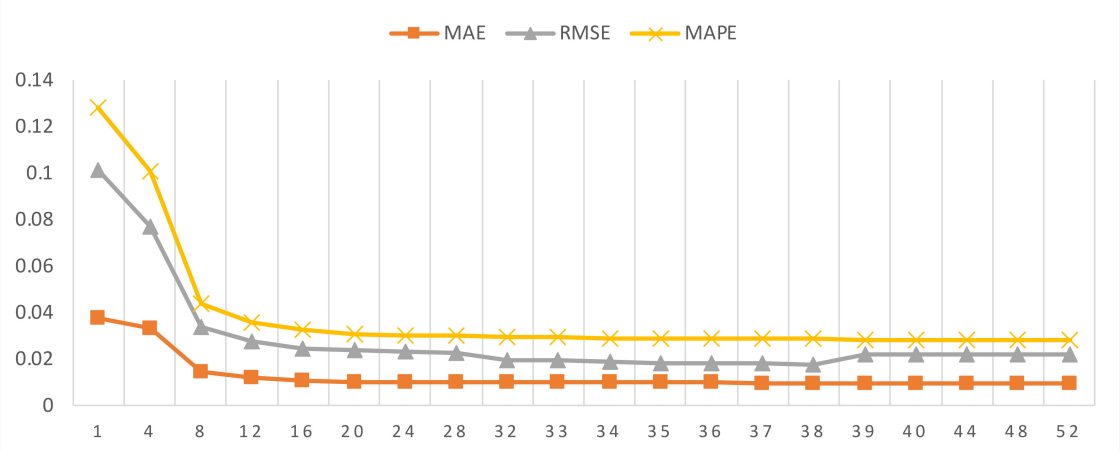


Figure 11. The impact of lag order on exchange rates for one-step-ahead forecasting.

From this figure, we can observe that the performance is significantly improved as the lag order increased from 1 to 8. Once the lag order is greater than 16, all these metrics would be stable. RMSE slightly increased with the increase of lag order from 39 to 52, while MAE and MAPE remained stable. These results tell us that the lag order plays an important role in improving forecasting performance.

4. Conclusions

In this paper, we proposed a CEEMDAN-based long short-term memory neural network that adopts the multilayer stacked architecture to forecast the trend of exchange rate. We denoted it as CEEMDAN–MLSTM. The main steps are the following: (1) The exchange is decomposed into several IMFs and one residue by CEEMDAN mode; (2) The IMFs and residue are input into the multilayer LSTM network for forecasts; (3) The prediction of exchange rate data is obtained by summing up the forecast results of each IMF and the residue. We compared the proposed method with ARIMA, Bayesian, SVM, RNN, MRNN, LSTM, MLSTM, and MLSTM–CEEMDAN. The experiment proves that the proposed model has a better predictive performance than the other single models or combined models. This new approach has the following characteristics: (1) CEEMDAN algorithm, which can deeply explore the fluctuation characteristics of different time-scales of exchange rate data; (2) CEEMDAN algorithm solves the problem of mode mixing in empirical mode decomposition and the problem of signal reconstruction error; (3) LSTM adds a gating mechanism based on the classical RNN, so that the model can learn the data dependencies with a longer time span without the problem of gradient disappearance and gradient explosion. LSTM is more suitable for solving exchange rate forecasting problems than RNN. (4) Empirical evidence shows that properly increasing the depth of the neural network will improve the ability of solving the complex problems.

In addition, this hybrid model of CEEMDAN–MLSTM was developed to solve the problems of short-term exchange rate forecasts. Therefore, the factors that affect exchange rate, such as interest rates, balance of payments, economic growth, inflation rates, etc., have not been taken into account. All forecasts are based on historical price data. If the long-term forecasts are required, we should combine our proposed method with economic theory or measurement methods to play a greater role.

In this study, the proposed CEEMDAN–MLSTM was used to improve the accuracy of exchange rate forecasts for reducing the risks of international trade. However, this does not mean that this method can only solve this particular issue. It also can be extended to solve more nonlinear and nonstationary time series problems, such as natural disaster prediction, energy price prediction, financial time series regression problems, risk prediction and decision, etc.

Author Contributions: Conceptualization, H.L.; Data curation, S.-Q.C.; Formal analysis, Q.S.; Investigation, Q.S.; Methodology, H.L. and S.-Q.C.; Writing–original draft, H.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (No. 71872046), Environmental Regulation, Enterprise Innovation and Industrial Transformation and Upgrade: Theory and Practice Based on Energy Saving and Emissions Supervision.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Brada, J.C.; Méndez, J.A. Exchange rate risk, exchange rate regime and the volume of international trade. *Kyklos* **2007**, *41*, 263–280. [CrossRef]
2. Korhonen, A. Strategic financial management in a multinational financial conglomerate: A multiple goal stochastic programming approach. *Eur. J. Oper. Res.* **2001**, *128*, 418–434. [CrossRef]
3. Takatoshi, I.; Satoshi, K.; Kiyotaka, S.; Junko, S. Exchange rate exposure and exchange rate risk management: The case of japanese exporting firms. *Discuss. Pap.* **2013**, *41*, 17–29.
4. Papaioannou, M.G. Exchange rate risk measurement and management: Issues and approaches for firms. *IMF Work. Pap.* **2006**, *6*, 1–20. [CrossRef]
5. Davidson, J.; Li, X. Strict stationarity, persistence and volatility forecasting in ARCH(∞) processes. *J. Empir. Financ.* **2016**, *38*, 534–547. [CrossRef]
6. Kristjanpoller, W.; Minutolo, M.C. Forecasting volatility of oil price using an artificial neural network-GARCH model. *Expert Syst. Appl.* **2016**, *65*, 233–241. [CrossRef]

7. Riesgo García, M.V.; Krzemień, A.; Manzanedo del Campo, M.Á.; Escanciano García-Miranda, C.; Sánchez Lasheras, F. Rare earth elements price forecasting by means of transgenic time series developed with ARIMA models. *Resour. Policy* **2018**, *59*, 95–102. [CrossRef]
8. Fan, J.; Yao, Q. Nonlinear Time Series: Nonparametric and Parametric Methods. *J. Am. Stat. Assoc.* **2005**, *100*, 348–349.
9. Box, G. Box and jenkins: Time series analysis, forecasting and control. In *A Very British Affair: Six Britons and the Development of Time Series Analysis during the 20th Century*; Mills, T.C., Ed.; Palgrave Macmillan UK: London, UK, 2013; pp. 161–215.
10. Engle, R.F. Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. *Econometrica* **1982**, *50*, 987–1007. [CrossRef]
11. Bollerslev, T. Generalized autoregressive conditional heteroskedasticity. *EERI Res. Pap.* **1986**, *31*, 307–327. [CrossRef]
12. Lin, L.; Fang, W.; Xie, X. Random forests-based extreme learning machine ensemble for multi-regime time series prediction. *Expert Syst. Appl. An Int. J.* **2017**, *83*, 164–176. [CrossRef]
13. Cao, L. Support vector machines experts for time series forecasting. *Neurocomputing* **2003**, *51*, 321–339. [CrossRef]
14. Yan, D.; Qi, Z.; Wang, J.; Na, Z. Bayesian regularisation neural network based on artificial intelligence optimisation. *Int. J. Prod. Res.* **2016**, *55*, 2266–2287. [CrossRef]
15. Rather, A.M.; Agarwal, A.; Sastry, V.N. Recurrent neural network and a hybrid model for prediction of stock returns. *Expert Syst. Appl.* **2015**, *42*, 3234–3241. [CrossRef]
16. 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. *Appl. Soft Comput.* **2011**, *11*, 2510–2525. [CrossRef]
17. Rosas-Romero, R.; Díaz-Torres, A.; Etcheverry, G. Forecasting of stock return prices with sparse representation of financial time series over redundant dictionaries. *Expert Syst. Appl.* **2016**, *57*, 37–48. [CrossRef]
18. Zhou, T.; Gao, S.; Wang, J.; Chu, C.; Todo, Y.; Tang, Z. Financial time series prediction using a dendritic neuron model. *Knowl. Based Syst.* **2016**, *105*, 214–224. [CrossRef]
19. Mammadli, S. Financial time series prediction using artificial neural network based on Levenberg-Marquardt algorithm. *Procedia Comput. Sci.* **2017**, *120*, 602–607. [CrossRef]
20. Kim, H.Y.; Won, C.H. Forecasting the volatility of stock price index: A hybrid model integrating LSTM with multiple GARCH-type models. *Expert Syst. Appl.* **2018**, *103*, 25–37. [CrossRef]
21. Guresen, E.; Kayakutlu, G.; Daim, T.U. Using artificial neural network models in stock market index prediction. *Expert Syst. Appl.* **2011**, *38*, 10389–10397. [CrossRef]
22. Bisoi, R.; Dash, P.K. A hybrid evolutionary dynamic neural network for stock market trend analysis and prediction using unscented Kalman filter. *Appl. Soft Comput.* **2014**, *19*, 41–56. [CrossRef]
23. Ma, Z.; Dai, Q.; Liu, N. Several novel evaluation measures for rank-based ensemble pruning with applications to time series prediction. *Expert Syst. Appl.* **2015**, *42*, 280–292. [CrossRef]
24. Gong, X.; Si, Y.-W.; Fong, S.; Biuk-Aghai, R.P. Financial time series pattern matching with extended UCR suite and support vector machine. *Expert Syst. Appl.* **2016**, *55*, 284–296. [CrossRef]
25. Pwasong, A.; Sathasivam, S. A new hybrid quadratic regression and cascade forward backpropagation neural network. *Neurocomputing* **2016**, *182*, 197–209. [CrossRef]
26. Wang, J.; Peng, B.; Zhang, X. Using a stacked residual LSTM model for sentiment intensity prediction. *Neurocomputing* **2018**, *322*, 93–101. [CrossRef]
27. EA-LSTM: Evolutionary attention-based LSTM for time series prediction. *Knowl. Based Syst.* **2019**, *181*, 104785. [CrossRef]
28. Liu, G.; Guo, J. Bidirectional LSTM with attention mechanism and convolutional layer for text classification. *Neurocomputing* **2019**. [CrossRef]
29. Sagheer, A.; Kotb, M. Time series forecasting of petroleum production using deep LSTM recurrent networks. *Neurocomputing* **2019**, *323*, 203–213. [CrossRef]
30. Dai, S.; Li, L.; Li, Z. Modeling vehicle interactions via modified LSTM models for trajectory prediction. *IEEE Access* **2019**, *7*, 38287–38296. [CrossRef]

31. Hao, X.; Du, Q.H.; Mark, R. SS-LSTM: A Hierarchical LSTM Model for Pedestrian Trajectory Prediction. In Proceedings of the IEEE Winter Conference on Applications of Computer Vision. IEEE, 2018, Lake Tahoe, NV, USA, 12–15 March 2018.
32. Tang, L.; Dai, W.; Yu, L.; Wang, S. A Novel CEEMD-Based EELM Ensemble Learning Paradigm for Crude Oil Price Forecasting. *Int. J. Inf. Technol. Decis. Mak.* **2015**, *14*, 141–169. [CrossRef]
33. Cao, J.; Li, Z.; Li, J. Financial time series forecasting model based on CEEMDAN and LSTM. *Phys. A Stat. Mech. Its Appl.* **2018**. [CrossRef]
34. Huang, N.E.; Shen, Z.; Long, S.R.; Wu, M.L.C.; Shih, H.H.; Zheng, Q.N.; Yen, N.C.; Tung, C.C.; Liu, H.H. The empirical mode decomposition and the Hilbert spectrum for nonlinear and nonstationary time series analysis. *Proc. R. Soc. A Math. Phys. Eng. Sci.* **1998**, *454*, 903–995. [CrossRef]
35. Wu, Z.; Huang, N. Ensemble empirical mode decomposition: A noise-assisted data analysis method. *Adv. Adapt. Data Anal.* **2009**, *1*, 1–41. [CrossRef]
36. Zhang, N.; Lin, A.; Shang, P. Multidimensional k-nearest neighbor model based on EEMD for financial time series forecasting. *Phys. A Stat. Mech. Its Appl.* **2017**, *477*, 161–173. [CrossRef]
37. Das, A.B.; Bhuiyan, M.I.H. Discrimination of focal and non-focal EEG signals using entropy-based features in EEMD and CEEMDAN domains. In Proceedings of the 9th International Conference on Electrical and Computer Engineering (ICECE), Dhaka, Bangladesh, 20–22 December 2016.
38. Pradeepkumar, D.; Ravi, V. Forecasting financial time series volatility using Particle Swarm optimization trained quantile regression neural network. *Appl. Soft Comput.* **2017**, *58*, 35–52. [CrossRef]
39. Meese, R.A.; Rogoff, K. Empirical exchange rate models of the seventies: Do they fit out of sample? *J. Int. Econ.* **1983**, *14*, 3–24. [CrossRef]
40. Galeshchuk, S. Neural networks performance in exchange rate prediction. *Neurocomputing* **2016**, *172*, 446–452. [CrossRef]
41. Wright, J.H. Bayesian model averaging and exchange rate forecasts. *J. Econom.* **2008**, *146*, 329–341. [CrossRef]
42. Jia, S.; Guo, Y.; Qiang, W.; Jian, Z. Trend extraction and similarity matching of financial time series based on emd method. In Proceedings of the World Congress on Computer Science & Information Engineering, Los Angeles, CA, USA, 31 March–2 April 2009.