Expert Systems With Applications 206 (2022) 117736

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Forecasting the realized volatility of stock price index: A hybrid model integrating CEEMDAN and LSTM

Yu Lin, Zixiao Lin *, Ying Liao, Yizhuo Li, Jiali Xu, Yan Yan

School of Business, Chengdu University of Technology, Chengdu 610059, China

ARTICLE INFO

Keywords:
Volatility forecasting
Realized volatility
LSTM
CEEMDAN
MCS test

ABSTRACT

The realized volatility (RV) financial time series is non-linear, volatile, and noisy. It is not easy to accurately forecast RV with a single forecasting model. This paper adopts a hybrid model integrating Long Short-Term Memory (LSTM) and complete ensemble empirical mode decomposition with adaptive noise (CEEMDAN) to forecast the RV of CSI300, S&P500, and STOXX50 indices. After the empirical study, four loss functions MSE, MAE, HMSE, HMAE, and the model confidence set (MCS) test are taken as the evaluation criteria. This paper selected Back Propagation Neural Networks (BP), Elman Neural Networks (Elman), Support Vector Regression Machine (SVR), autoregression (AR), heterogeneous autoregressive (HAR), and their hybrid models with CEEMDAN as the comparison. The test results show that CEEMDAN-LSTM has the best performance in forecasting RV in emerging and developed markets. Besides, the performance of single models is inferior to their corresponding hybrid models with CEEMDAN. And the empirical results are robust with the "sliding window" approach.

1. Introduction

Volatility is not only a standard measure of risk in the financial market, but also plays an essential role in asset pricing and portfolio selection. How to accurately forecast volatility has always been a hot topic of research. In the early years, scholars generally concentrated the forecast of low-frequency volatility (Engle, 1982; Bollerslev, 1986; Kat & Heynen, 1994). However, the forecast results of low-frequency volatility cannot be a suitable reference for future market risks owing to the massive noise that low-frequency volatility inevitably contains (Andersen & Bollerslev, 1998). Andersen and Bollerslev (1998) made the seminal contribution for proposing realized volatility/ realized variance[1] (RV) constituted by high-frequency data. RV has become an essential way of measuring actual volatility since the advantages of more information, less noise, unbiasedness, and robustness (Andersen, Bollerslev, & Diebold, 2002).

When RV was first proposed, a great range of researches on it was conducted around the low-frequency volatility models. Such as autoregression (AR) (Andersen, Bollerslev, Diebold, & Labys, 2003), autoregressive fractionally integrated moving average (ARFIMA) (Takahashi, Omori, & Watanabe, 2009), and stochastic volatility (SV) (Wei, Wang, & Huang, 2010). Subsequently, more models were explicitly constructed for RV forecasting, such as the popular heterogeneous autoregressive model of realized volatility (HAR) (Corsi, 2009). HAR is simple in form and has the advantage of capturing "stylized facts," such as long memory and multi-behavior (Ma, Wahab, Huang, & Xu, 2017). Thereby, many scholars have conducted researches based on HAR and its extended models (Ma, Liu, Huang, & Chen, 2017 Chen, Ma, Wei, & Liu, 2020). Hillebrand and Medeiros (2010) presented a linear extension of the HAR model with exogenous variables and shows that bagging can improve forecasting accuracy. Duan, Chen, Zeng, and Liu (2018) investigated in the framework of regime-switching how leverage effect and economic policy uncertainty impact the RV of S&P500. Luo, Ji, Klein, Ned, and Zhang (2020) found that the Infinite Hidden Markov-HAR model performs better in crude oil futures markets than other non-switching variants of HAR.

In recent years, with the rapid development of artificial intelligence, deep learning algorithms have been extensively applied in various fields (Lu, Que, & Cao, 2016; Kodama, Pichl, & Kaizoji, 2017). Compared with traditional econometric models, deep learning algorithms are superior with less restriction and the strength of feature extraction (Kim & Won, 2018). Thus, scholars began to explore the feasibility of applying deep

* Corresponding author.
E-mail address: linzixiao0123@163.com (Z. Lin).

[1] In this paper the terms realized volatility and realized variance will be used interchangeably.

https://doi.org/10.1016/j.eswa.2022.117736
Received 28 April 2020; Received in revised form 11 January 2021; Accepted 31 May 2022
Available online 3 June 2022

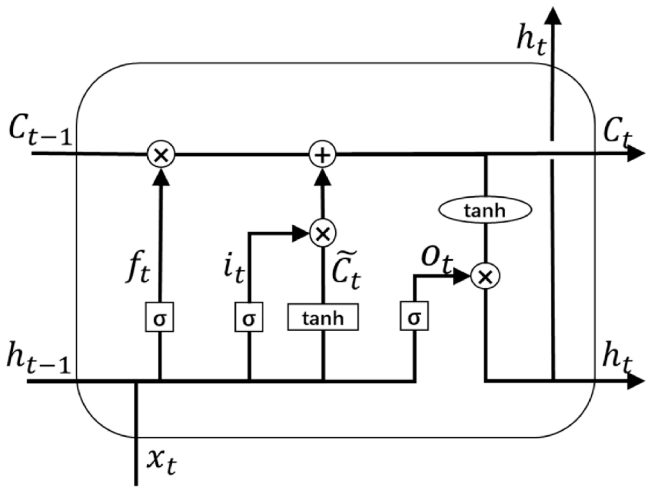

Fig. 1. One basic unit of LSTM.

learning in RV forecasting. McAleer and Medeiros (2011) introduced a non-linear HAR model based on neural networks. Arnerić, Poklepović, and Teai (2018) compared two approaches: HAR and Feedforward Neural Networks (FNN). Moreover, they concluded that FNN-HAR-type models could better capture the non-linear characteristic of RV.

Distinct from FNN, there are loop structures in Recurrent Neural Networks (RNN) that allow information persistence. Hence RNN is more suitable for processing financial time series than FNN (Cao, Li, & Li, 2019). Andersen, Bollerslev, Diebold and Labys (1999) have confirmed the long-memory feature of RV. As an improved variant of RNN, LSTM (Hochreiter and Schmidhuber, 1997) can capture the features of time series over a longer time range (Zhao, Chen, Wu, Chen, & Liu, 2017); theoretically, LSTM is suitable for RV forecasting. Bucci (2019) and Miura, Pichl, and Kaizoji (2019) compared RV forecasting performance of traditional econometric models and deep learning algorithms where LSTM is included in the cryptocurrency market. Nevertheless, there is a mixed result of whether LSTM has the best forecast performance, which could be caused by RV's noise (Zhang, Mykland, & Ait-Sahalia, 2005).

Previous studies have found that decomposition can separate the noise part from the original time series, thus, improve the forecasting accuracy (Premanode and Toumazou, 2013; Cao et al., 2019). Accordingly, selecting an appropriate decomposition method is of importance. Although empirical mode decomposition (EMD) (Huang, Shen, Long, Wu, & Shih, 1998) could processes data with high complexity and irregularity, it still has a flaw: mode mixing. Ensemble empirical mode decomposition (EEMD) (Wu & Huang, 2009) was proposed to fix mode mixing by adding Gaussian white noise to the original series. However, the addition of white noise also raises a new problem: the reconstruction error cannot be eliminated. Therefore, Torres, Colominas, Schlotthauer, and Flandrin (2011) proposed CEEMDAN to solve this problem.

As a consequence, in this paper, a novel hybrid model CEEMDAN-LSTM is adopted for RV forecasting. We will compare the performance of CEEMDAN-LSTM with another 10 models under MSE, MAE, HMSE, and HMAE criteria. To obtain more robust test results, the model confidence set (MCS) (Hansen, Lunde, & Nason, 2011) test is employed. Furthermore, the forecasting performance of CEEMDAN-LSTM in both emerging markets and developed markets needs to be testified. Accordingly, we take Shanghai and Shenzhen 300 Index (CSI300), Standard & Poor's 500 Index (S&P500), and Euro Stoxx 50 Index (STOXX50) as the representatives of the stock markets in China, American, and Europe, respectively. In general, this paper aims to answer the following two questions through empirical study: (1) Whether CEEMDAN-LSTM can accurately forecast RV or not? (2) Supposing that the last question could be answered in the affirmable, if CEEMDAN-LSTM has the best performance in emerging markets and developed markets?

This paper's remainder is organized as follows: Section 2 provides a general description of CEEMDAN and LSTM and carries out the algorithm designing. Section 3 describes the data and provides a preliminary analysis. Section 4 carries out empirical research and the analyses of results. At last, Section 5 concludes.

2. Methodology

2.1. CEEMDAN

In this study, we use the intraday stock index return to construct the daily RV. For a given day t, $RV(t)$ is the summation of intraday high-frequency squared returns:

$$RV(t) = \sum_{i=1}^{1/\Delta} {r_{t,i}}^2 \tag{1}$$

Where Δ is the sampling interval and $1/\Delta$ is the sampling number, $r_{t,i}$ represents the i-th intraday return on day t.

To reduce the noise's impact on RV forecasting, the original series $RV(t)$ are decomposed. EMD assumes that every series is composed of several Intrinsic Mode Functions (IMF) and one residue. IMFs must satisfy the following two conditions: (1) in the whole data set, the number of extrema (both maxima and minima) equals the number of zero crossings with the tolerance of one; (2) at any point, the envelope defined by the local maxima (upper envelope) and the envelope defined by the local minima (lower envelope) must be subject to the condition of zero mean.

To some extent, EEMD overcomes the mode mixing of EMD by adding Gaussian white noise to the original series. Furthermore, compared with EEMD, CEEMDAN almost eliminate reconstruction errors. The steps of CEEMDAN are as follows:

(1). Add white noise $\varepsilon_0\omega_i(t)$ to the original series $RV(t)$ to produce M different new series, the white noise $\varepsilon_0\omega_i(t)$ $N(0, \varepsilon_0)$. Then decompose each new series through EMD. Thus, the first IMF and the first residue of CEEMDAN is obtained:

$$IMF_1(t) = M^{-1}\sum_{i=1}^{M} E_1(RV(t) + \varepsilon_0\omega_i(t)) \tag{2}$$

$$R_1(t) = RV(t) - IMF_1(t) \tag{3}$$

The operator $E_1()$ is defined as the first IMF obtained by EMD.

(2). For the following $N-1$ IMFs of CEEMDAN, the process is a little different: first, add white noise $\varepsilon_{j-1}E_{j-1}(\omega_i(t))$ to the residue $R_{j-1}(t)$ to produce M different new residues. Then these new residues are decomposed through EMD to obtain the j-th IMF and the j-th residue of CEEMDAN:

$$IMF_j(t) = M^{-1}\sum_{i=1}^{M} E_1\left(R_{j-1}(t) + \varepsilon_{j-1}E_{j-1}(\omega_i(t))\right), j = 2, \cdots, N \tag{4}$$

$$R_j(t) = R_{j-1}(t) - IMF_j(t) \tag{5}$$

(3). Repeat step 2 until $R_j(t)$ is no longer feasible to be decomposed ($R_j(t)$ does not have at least two extrema). Finally, the residue of CEEMDAN is obtained:

$$R(t) = RV(t) - \sum_{j=1}^{N} IMF_j(t) \tag{6}$$

2.2. Long short-term memory

The application of three "gates" and the cell state make it easy for LSTM to reset, update, and keep long-term information. In Fig. 1, the horizontal line running through the top is the cell state C_t, whereas the

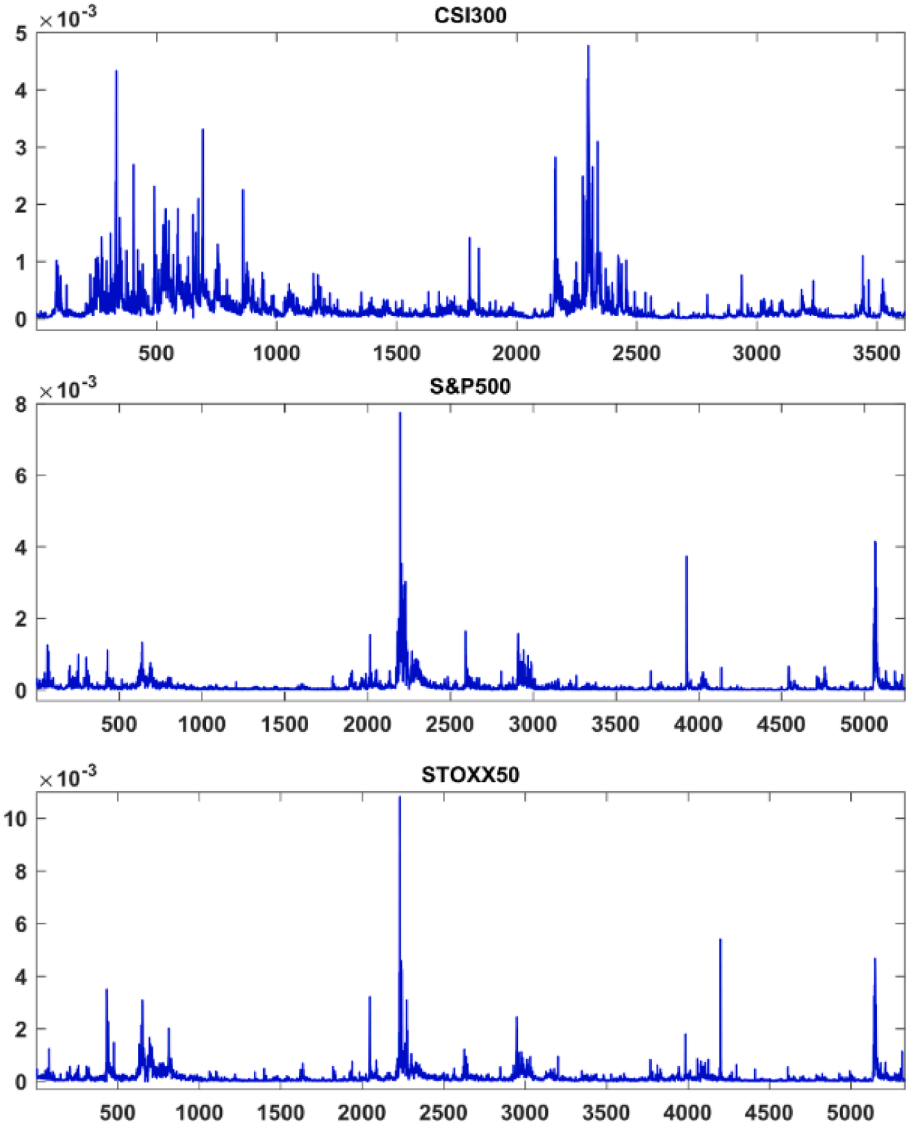

Fig. 2. The series of RV.

Table 1
Descriptive statistics of RV for stock indices.

Index	CSI 300	S&P 500	STOXX 50
Count	3615	5244	5331
Mean	1.99×10^{-4}	1.12×10^{-4}	1.61×10^{-4}
Standard deviation	3.18×10^{-4}	2.66×10^{-4}	3.32×10^{-4}
Skewness	5.9188***	10.7481***	12.1129
Excess Kurtosis	53.9905***	1.96×10^{2}***	2.64×10^{2}***
J-B	4.60×10^{5}***	8.21×10^{6}***	1.53×10^{7}***
Q(10)	8.26×10^{3}***	1.68×10^{4}***	1.15×10^{4}***

Note: J-B denotes the Jarque-Bera Statistics. Q(10) is the Ljung–Box statistic for up to 10th order serial correlation. *, ** and *** denote rejection of the null hypothesis at the 10%, 5%, 1% significance level, respectively.

bottom one is the hidden state h_t. x_t is the input of LSTM at day t. The process of an LSTM unit is shown as follows:

(1). The first step is to decide which part of the previous cell state C_{t-1} will be forgotten. And the decision can be made by the "forget gate layer":

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{7}$$

Where W_f is the weight matrix, b_f is the bias, the operator $\sigma(\,)$ is defined as a sigmoid layer that pushes the values between 0 (completely forget) and 1 (completely keep).

(2). The next step is the update of the cell state. The "input gate layer" will decide which part of the candidate cell state $\widetilde{C}_t$ can be updated to the new cell state C_t:

$$\widetilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{8}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{9}$$

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C}_t \tag{10}$$

The operator *tanh*() is defined as a tanh layer that pushes the values between −1 and 1. And "*" represents the Hadamard product.

(3). The last step is the output, where the "output gate layer" plays its role:

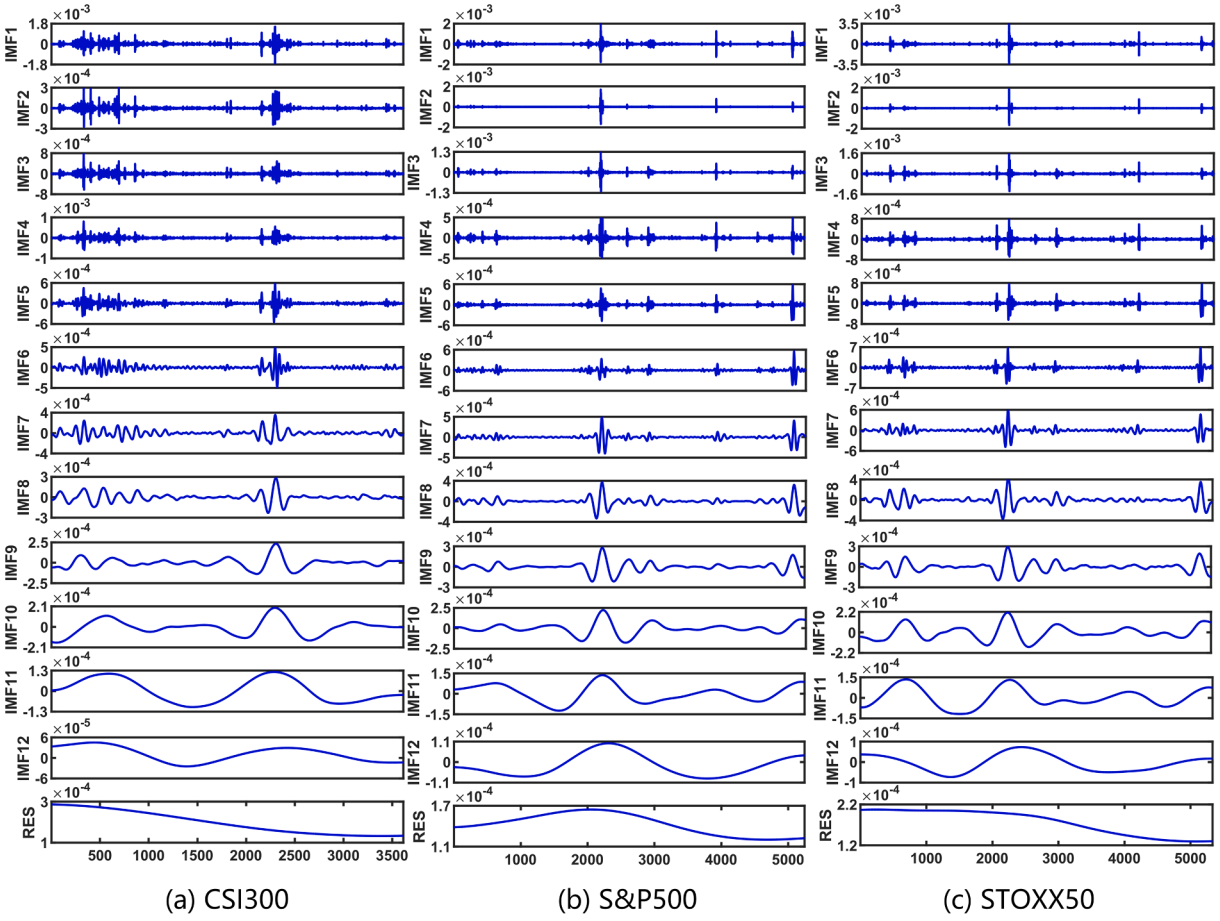

Fig. 3. Decomposition results of RV for stock indices.

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{11}$$

$$h_t = o_t{}^{*}tanh(C_t) \tag{12}$$

Algorithm design.

The structure of LSTM applying in this paper is as follows: the first layer is the input layer, the input at day t is $X_t = (x_{t-w+1}, x_{t-w+2}, \cdots, x_t)$, and the corresponding label is x_{t+1}. The neuron number of the following two LSTM layers is set as 128 and 64, respectively. The second LSTM layer is connected to a full connection layer with 16 neurons; then, the full connection layer will connect to the output layer, which has only one neuron. The activation function of all the layers is the ReLU (rectified linear unit) function as it has a faster calculation speed compared with the sigmoid function and the tanh function; it also resolves the gradient vanishing problem in the positive interval. To prevent over-fitting, dropout (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014) is adopted, and the parameter of it is set as 0.5. Moreover, the mean square error (MSE) is adopted as the loss function:

$$LOSS = MSE = H^{-1}\sum_{i=1}^{H}(\hat{y}_i - y_i)^2 \tag{13}$$

Where H is the length of the training set, y_i is the actual value of the i-th data and $\hat{y}_i$ is the predicted value of the i-th data.

Before the forecasting through LSTM, the original series $RV(t)$ is decomposed by CEEMDAN. As shown in Algorithm 1, our principal algorithm is designed as composed of three steps:

(1). Through CEEMDAN the original series $RV(t)$ is decomposed into $N+1$ sub-series: $IMF_i(t), i = 1, 2, \cdots, N$, and one residue $R(t)$.

(2). In this paper, we adopt "sliding window" as the robustness check method to test the model's forecasting performance on different data sets. Thus, in the empirical part, we only adopt the first 90% data of the whole data set. Then each sub-series are divided into a training set, a validation set, and a test set by 7:1:1. The training set data are then put into their corresponding LSTM forecasting models as the input to train; the validation set data are used to test the model's generalization performance; at last, the test set data are used to obtain the predicted results. The predicted results of test set are $\widehat{IMF_i}(t), i = 1, 2, \cdots, N$ and $\widehat{R(t)}$.

(3). To generate the final predicted series, all predicted results are reconstructed according to the following equation.

$$\widehat{RV(t)} = \sum_{i=1}^{N}\widehat{IMF_i}(t) + \widehat{R(t)}, t = H+V+1, H+V+2, \cdots, H+V+D \tag{14}$$

Where D is the length of the test set, V is the length of the validation set, H is the length of the training set, and $\widehat{RV(t)}$ is the final predicted series of the original series $RV(t)$.

Algorithm 1: CEEMDAN-LSTM

Input: original time series $RV(t)$, white noise $\varepsilon_0\omega_i(t)$, the ensemble number M

Output: predicted time series $\widehat{RV(t)}$

Function $CEEMDAN\ (RV(t), \varepsilon_0\omega_i(t), M)$
  $R_0(t) = RV(t)$
  $j = 1$
  while extrema in $R_j(t) \geq 2$ **do**
    $IMF_j(t) = []$
    for $k = 1{:}M$ **do**
      $IMF_j(t) = IMF_j(t) + E_1(R_{j-1}(t) + \varepsilon_{j-1}E_{j-1}(\omega_i(t)))$
    end for
    $IMF_j(t) = M^{-1}IMF_j(t)$
    $R_j(t) = R_{j-1}(t) - IMF_j(t)$
    $j = j + 1$
  end while
  $R(t) = R_j(t)$
return $IMF_j(t),\ R(t), j$
Function $LSTM\ (IMF_j(t)$ or $R(t))$
  $data = IMF_j(t)$ or $R(t)$
  $len = length\ (data)$
  $data\_train = data\ (1{:}(7/10 * len))$
  $data\_validation = data\ ((7/10 * len + 1){:}(8/10 * len))$
  $data\_test = data\ ((8/10 * len + 1){:}(9/10 * len))$
  $trained\_model = train\ (initial\_model, data\_train)$ % the model with the lowest MSE on the validation set is adopted.
  $data\_forecsat = forecast\ (trained\_model, data\_test)$
  $\widehat{IMF_i(t)}\ or\ \widehat{R(t)} = data\_forecast$
return $\widehat{IMF_i(t)}\ or\ \widehat{R(t)}$
$[IMF_j(t), R(t), j] = CEEMDAN\ (RV(t), \varepsilon_0\omega_i(t), M)$
for $i = 1{:}j$ **do**
  $\widehat{IMF_i(t)} = LSTM(IMF_i(t))$
end for
$\widehat{R(t)} = LSTM(R(t))$
for $i = 1{:}j$ **do**
  $\widehat{R(t)} = \widehat{R(t)} + \widehat{IMF_i(t)}$
end for
$\widehat{RV(t)} = \widehat{R(t)}$

2.3. Evaluation criteria

This study selects four common loss functions. L1 and L2 are the mean squared error (MSE) and the mean absolute error (MAE) respectively, L3 and L4 are heteroskedastic adjusted MSE and MAE, which are as follows:

$$\text{L1}: MSE = D^{-1}\sum_{t=H+V+1}^{H+V+D}(RV(t) - \widehat{RV(t)})^2 \tag{15}$$

$$\text{L2}: MAE = D^{-1}\sum_{t=H+V+1}^{H+V+D}|RV(t) - \widehat{RV(t)}| \tag{16}$$

$$\text{L3}: HMSE = D^{-1}\sum_{t=H+V+1}^{H+V+D}\left(1-\frac{\widehat{RV(t)}}{RV(t)}\right)^2 \tag{17}$$

$$\text{L4}: HMAE = D^{-1}\sum_{t=H+V+1}^{H+V+D}\left|1-\frac{\widehat{RV(t)}}{RV(t)}\right| \tag{18}$$

In the equations above D is the length of the test set, V is the length of the validation set, H is the length of the training set, $RV(t)$ is the actual RV value, $\widehat{RV(t)}$ is the predicted RV value.

Loss functions cannot provide any statistical information about whether the models' forecasting performance is significantly different. Therefore, the MCS test is adopted, hoping to obtain more robust test results. It enables direct comparison of models, and the process of it is as follows:

There is a total of n models to be tested. These models form the original set M^0. When the loss function is selected, let $L_{i,t}$ be the loss function value of the i-th model on day t. So that the relative performance variables in the MCS test of any two models in the M^0 can be calculated: $d_{ij,t} = L_{i,t} - L_{j,t}$, $\mu_{ij} = E(d_{ij,t})$, $i,j \in M^0$.

Define the set of superior objects as M^*, which has to meet the conditions: $M^* \equiv \{i \in M^0, \mu_{ij} \le 0, \forall j \in M^0\}$. That is, all the models in M^* have the same or better performance than each model in M^0, and M^* is the result that the MCS test finally pursues.

The MCS test follows two basic rules, namely δ_M (equivalence test) and e_M (elimination rule). The null hypothesis of δ_M is that the performance of the two models are equivalent; when the null hypothesis is rejected, the e_M is adopted to remove the model that is significantly inferior to the other models from M^0, thus obtaining the set $M \in M^0$. Repeat this process. Finally, when the null hypothesis of δ_M is no longer rejected, the set $\widehat{M^*_{1-\alpha}} = M$ is obtained, where α is the given significance level.

There are many T-statistic proposed for the MCS test. In this study, we adopt two of the most common ones: range statistic (T_R) and semi-quadratic statistic (T_{SQ}).

$$T_R = \max_{i,j\in M}\frac{|\overline{d_{ij}}|}{\sqrt{var(\overline{d_{ij}})}} \tag{19}$$

$$T_{SQ} = \max_{i,j\in M}\frac{(\overline{d_{ij}})^2}{var(\overline{d_{ij}})} \tag{20}$$

$$\overline{d_{ij}} = D^{-1}\sum_{t=H+V+1}^{H+V+D} d_{ij,t} \tag{21}$$

3. Data and preliminary analysis

Since 5-min high-frequency data can balance the requirements for accuracy and reduce micro noise (Andersen & Bollerslev, 1998), we take 5-min as the sampling interval. The sample data consists of intraday high-frequency price data of CSI300 index from August 8, 2005 to December 1, 2020; calculated RV data of S&P500 index from January 3, 2000 to December 1, 2020; and calculated RV data of STOXX50 index from January 3, 2000 to December 1, 2020. After the days with shortened trading sessions or too few transactions are removed, the RV of the CSI300 index can be obtained by Eq. (1). The price data of the CSI300 index comes from the Wind Database; the calculated RV data of the S&P500 and STOXX50 indices come from the Oxford-Man Institute's realised library.

Fig. 2 depicts RV series of CSI300, S&P500, and STOXX50 indices, distinct features of time-varying mean and variance are displayed. Moreover, there are significant differences between different markets: as the representative of emerging markets, the CSI300 index fluctuates dramatically over a longer time range than the S&P500 and the STOXX indices; moreover, the maximum RV value of CSI300 is smaller than the other two indices.

Table 1 exhibits the descriptive statistics of RV about three indices. The Jarque–Bera statistics show the rejection of null hypothesis at the 1% significance level, implying the abnormal distribution of these three series. And the Ljung–Box statistics show that the null hypothesis of no autocorrelation up to the 10th order is rejected at the 1% significance level, revealing the existence of a long-memory feature. The values of excess kurtosis evidence the stylized fact of a leptokurtic and fat-tailed distribution, and these two series are significantly right-skewed at the 1% significance level.

The original series $RV(t)$ of CSI300, S&P500, and STOXX50 indices are respectively decomposed through CEEMDAN. Fig. 3 shows the decomposition results. All IMFs and one residue are arranged from high frequency to low frequency, and all IMFs are fluctuating around zero, which makes them more predictable. The sub-series above represent high-frequency or noisy parts in the original series, which is the most difficult part to forecast. We can tell that RV indeed consists of massive noise as the first IMF's order of magnitude is the largest. And the residue at the bottom represents the trend of the original series. From the descriptive statistics we can claim that for all sub-series discussed in Table 2, standard deviation is smaller, and the statistical results of Ljung–Box test are significant at the 1% significance level, confirming serial correlation in these sub-series.

4. Empirical results

4.1. Hyperparameter selection

After decomposing, preprocessing of data is required to speed up the training process of LSTM. Applying standardization can also avoid the effects of outliers and extrema, and the formula for standardization is:

$$x_{std} = \frac{x - \text{mean}(x)}{\text{std}(x)} \tag{22}$$

Where x_{std} is the standardized data, $\text{mean}(x)$ is the mean of series x, and $\text{std}(x)$ is the standard variance of series x. After forecasting, the predicted values can be restored by the following formula:

$$\widehat{x_t} = \widehat{x_t'} * \text{std}(x) + \text{mean}(x) \tag{23}$$

Where $\widehat{x_t'}$ is the output of the forecasting model.

To strike a balance of less computation and better forecasting performance, two hyperparameters are set as fixed values: the learning rate is set as 0.001, and the batch size (mini-batch) is set as 32; two hyperparameters: "Window" and "Epoch" are adjusted through experiments. And the hyperparameters with the lowest MSE value of the validation set are adopted.

The optimal hyperparameter sets are displayed in Table 3. For these sub-series, the lower frequency is, the less complicate the sub-series is, and the more prominent the feature of long-memory is, thus wider "Window" and less "Epoch" are required. Moreover, we can see that for the last few sub-series, "Window" is getting narrower. Because for a simple time series, too much training data could cause "over-fitting"; thus, the model's generalization performance would decline.

Table 2
Descriptive statistics of the decomposition results of RV.

Component	Mean	Standard Deviation	Skewness	Excess Kurtosis	Q(10)
Panel A: CSI 300					
IMF1	-1.78×10^{-6}	1.32×10^{-4}	0.6322***	38.1805***	9.65×10^{2}***
IMF2	-7.87×10^{-8}	$2.59 \times 10^{--5}$	0.9832***	42.4811***	4.22×10^{3}***
IMF3	-1.50×10^{-7}	5.96×10^{-5}	0.6760***	30.2707***	5.92×10^{3}***
IMF4	-6.25×10^{-7}	7.47×10^{-5}	0.7952***	20.8977***	1.06×10^{4}***
IMF5	-5.82×10^{-7}	7.07×10^{-5}	0.3789***	17.0963***	1.31×10^{4}***
IMF6	-2.53×10^{-7}	7.08×10^{-5}	0.3994***	13.2310***	1.97×10^{4}***
IMF7	-1.48×10^{-6}	6.81×10^{-5}	0.7244***	4.3585***	3.10×10^{4}***
IMF8	-3.48×10^{-7}	6.02×10^{-5}	0.2959***	4.8421***	3.44×10^{4}***
IMF9	-2.53×10^{-6}	6.06×10^{-5}	0.7625***	2.8855***	3.59×10^{4}***
IMF10	-4.81×10^{-6}	7.72×10^{-5}	0.1914***	0.1146	3.59×10^{4}***
IMF11	3.70×10^{-6}	7.25×10^{-5}	0.1962***	−1.3980***	3.62×10^{4}***
IMF12	1.12×10^{-5}	2.17×10^{-5}	-5.91×10^{-2}	−1.3155***	3.61×10^{4}***
RES	1.97×10^{-4}	5.45×10^{-5}	0.2791***	−1.4336***	3.60×10^{4}***
Panel B: S&P 500					
IMF1	-4.22×10^{-7}	9.53×10^{-5}	−0.1755***	1.32×10^{2}***	3.03×10^{3}***
IMF2	-1.35×10^{-7}	5.83×10^{-5}	0.6476***	4.25×10^{2}***	5.05×10^{3}***
IMF3	-2.79×10^{-7}	6.13×10^{-5}	−0.9961***	1.55×10^{2}***	1.11×10^{4}***
IMF4	-2.27×10^{-7}	4.63×10^{-5}	0.1391***	46.4416***	1.46×10^{4}***
IMF5	-7.37×10^{-7}	5.34×10^{-5}	0.2482***	41.7062***	1.75×10^{4}***
IMF6	-1.04×10^{-6}	5.55×10^{-5}	-3.00×10^{-3}	30.5360***	3.02×10^{4}***
IMF7	-9.75×10^{-7}	6.97×10^{-5}	-3.38×10^{-2}	17.7455***	4.43×10^{4}***
IMF8	-4.75×10^{-6}	7.53×10^{-5}	6.73×10^{-3}	7.7749***	5.02×10^{4}***
IMF9	-3.11×10^{-6}	6.79×10^{-5}	8.16×10^{-2}**	3.8373***	5.16×10^{4}***
IMF10	-1.71×10^{-6}	6.80×10^{-5}	0.1610***	1.8337***	5.22×10^{4}***
IMF11	1.53×10^{-6}	6.22×10^{-5}	4.68×10^{-2}	−0.6820***	5.24×10^{4}***
IMF12	-1.90×10^{-5}	5.72×10^{-5}	0.6838***	−0.7485***	5.25×10^{4}***
RES	1.43×10^{-4}	1.54×10^{-5}	−0.1354***	−1.3854***	5.24×10^{4}***
Panel C: STOXX 50					
IMF1	-1.08×10^{-6}	1.41×10^{-4}	−1.0344***	2.15×10^{2}***	2.15×10^{3}***
IMF2	-1.53×10^{-7}	5.27×10^{-5}	−0.2229***	6.03×10^{2}***	4.72×10^{3}***
IMF3	-5.95×10^{-7}	8.01×10^{-5}	−0.3470***	1.25×10^{2}***	8.25×10^{3}***
IMF4	-3.25×10^{-7}	6.07×10^{-5}	0.5312***	43.9700***	1.52×10^{4}***
IMF5	-8.77×10^{-7}	7.18×10^{-5}	0.7919***	35.5578***	1.76×10^{4}***
IMF6	-1.10×10^{-6}	8.03×10^{-5}	0.3261***	23.0520***	2.84×10^{4}***
IMF7	-1.37×10^{-6}	8.34×10^{-5}	0.2557***	16.2846***	4.43×10^{4}***
IMF8	-3.04×10^{-6}	8.78×10^{-5}	0.3488***	6.0729***	5.09×10^{4}***
IMF9	-1.35×10^{-6}	7.27×10^{-5}	0.3518***	2.7268***	5.25×10^{4}***
IMF10	-2.45×10^{-6}	7.45×10^{-5}	0.4279***	0.1112**	5.31×10^{4}***
IMF11	1.79×10^{-6}	6.79×10^{-5}	0.2662***	−0.7247***	5.32×10^{4}***
IMF12	-5.47×10^{-6}	4.12×10^{-5}	0.2497***	−1.0299***	5.33×10^{4}***
RES	1.77×10^{-4}	2.97×10^{-5}	−0.4729***	−1.4507***	5.32×10^{4}***

Note: Q(10) is the Ljung–Box statistic for up to tenth order serial correlation. *, ** and *** denote rejection of the null hypothesis at the 10%, 5%, 1% significance level, respectively.

Table 3
Hyperparameters of CEEMDAN-LSTM.

	CSI 300		S&P 500		STOXX 50	
	Window	Epoch	Window	Epoch	Window	Epoch
IMF1	3	420	3	490	3	420
IMF2	3	420	3	430	4	390
IMF3	3	390	3	420	4	360
IMF4	4	380	4	410	4	360
IMF5	4	360	5	380	5	340
IMF6	5	360	5	360	5	300
IMF7	5	340	5	320	6	250
IMF8	5	290	6	270	6	230
IMF9	5	280	5	250	6	170
IMF10	4	230	5	210	5	160
IMF11	1	220	4	180	3	140
IMF12	1	170	3	170	1	70
Residue	1	110	1	60	1	60

Note: The “Window” is the length of each input. The “Epoch” is the number of training.

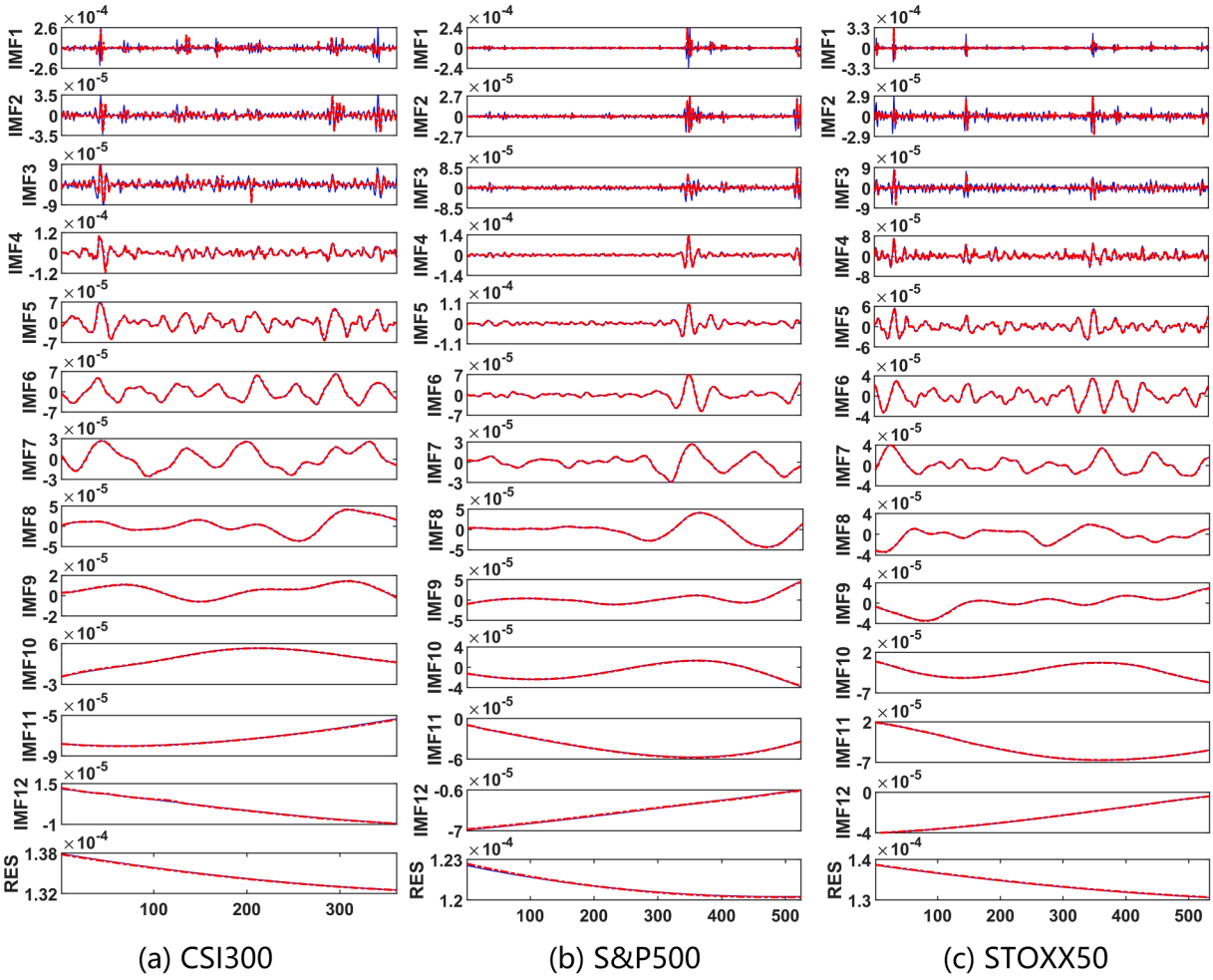

Fig. 4. Forecasting results of sub-series for S&P500 and CSI300 indices.

Fig. 4 shows the forecasting results of sub-series for CSI300, S&P500, and STOXX50 indices. The solid blue line represents the actual values, whereas the dashed red line represents the predicted values. The higher frequency is, the less smooth and predictable sub-series is. Accordingly, the high-frequency sub-series' accuracy is relatively low, whereas the low-frequency sub-series' predicted values are closer to the actual values.

4.2. Forecasting performance

In this study, LSTM is conducted in Python 3.6 via TensorFlow 1.4.0 (CPU); the other models are conducted in MATLAB R2019a; the MCS test results are obtained through OxMetrics 6.0. All work is running on a computer with a 3.5 GHz CPU and 8 GB RAM. After obtaining the predicted values of each sub-series, the final predicted values of RV can be obtained by Eq. (14), and the results are shown in Fig. 5.

Besides LSTM, we select other deep learning algorithms and traditional econometric models as the comparison, including BP, ELMAN, SVR, HAR, and AR. We also processed forecasting through these models on the sub-series decomposed by CEEMDAN. Furthermore, on account of the negative values in sub-series that make weekly and monthly RV in HAR meaningless, we did not adopt HAR to forecast the sub-series.

Table 4 reports the forecasting error of these 11 models. For the CSI300 index, CEEMDAN-LSTM has the lowest values under MSE, MAE, HMSE, and HMAE, which indicates that CEEMDAN-LSTM has the best forecast performance. CEEMDAN-LSTM also performs the best forecasting S&P500 index. Moreover, the performance of hybrid models with CEEMDAN is better than their corresponding single models, proving decomposition's effectiveness. Besides, we can tell that the HMSE values of STOXX50 index are remarkably large. This happened because the 407th actual RV value is 9.29×10^{-5}; it is tiny compared to the rest actual RV values. Thus, a little forecasting error would lead to a sizeable HMSE value.

In addition, assuming there are three kinds of investors: short-term, middle-term, and long-term investors, we evaluated the forecasting performance of models over different time horizons with three different time windows: 5 days (one week), 21 days (one month), and 252 days (one year). Table 5 exhibits the forecasting evaluation results over different time horizons. We can tell that except for 5-window length and 21-window length of the CSI300 index, CEEMDAN-LSTM still performs the best, which confirms the superiority of CEEMDAN-LSTM over different time horizons.

To further demonstrate the superiority of CEEMDAN-LSTM, the MCS test is adopted. We perform 20,000 bootstraps with the length of 5 blocks to generate *p*-values of the MCS test, and the significance level of the MCS test is set as 25%, which means that only the model with a *p*-

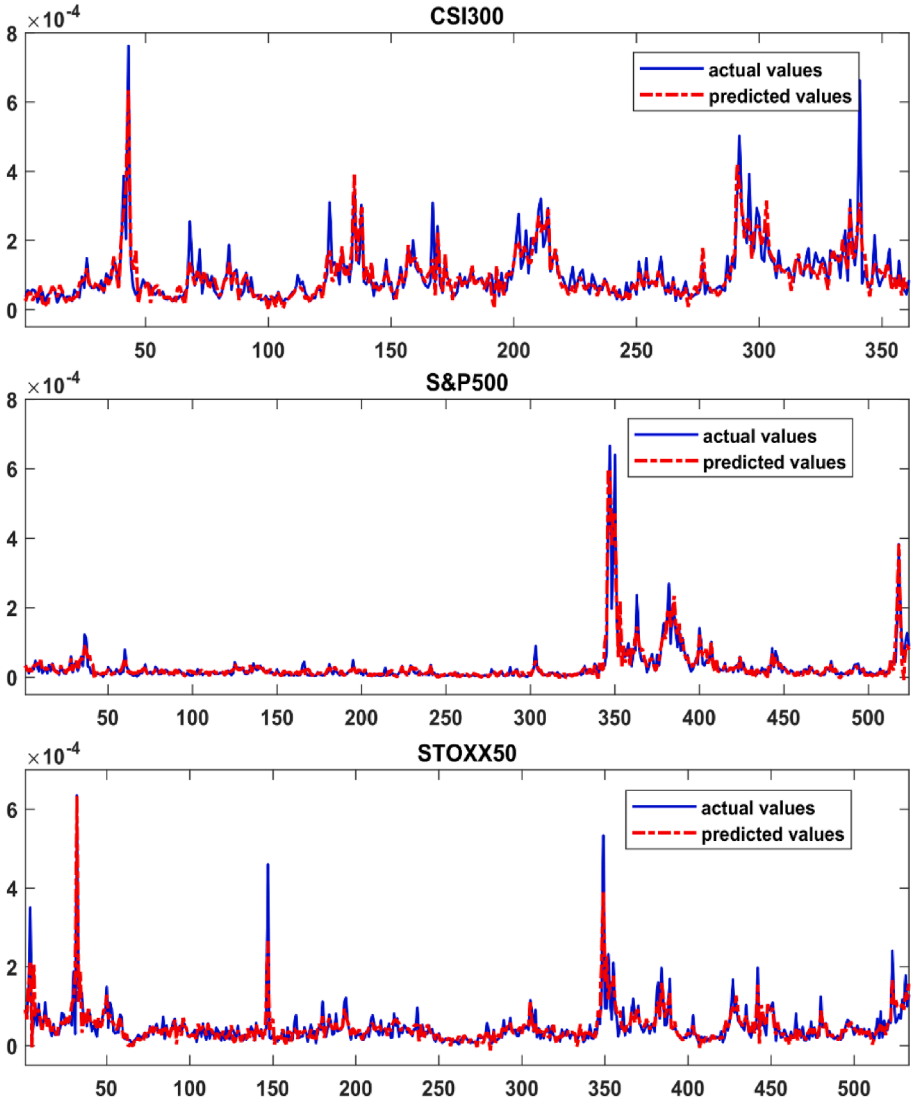

Fig. 5. Forecasting results of RV through CEEMDAN-LSTM.

Table 4
Forecasting evaluation results under four loss functions.

Models	MSE	MAE	HMSE	HMAE	MSE	MAE	HMSE	HMAE	MSE	MAE	HMSE	HMAE
	Panel A: CSI 300				Panel B: S&P 500				Panel C: STOXX 50			
CEEMDAN-LSTM	$\mathbf{1.54 \times 10^{-9}}$	$\mathbf{2.39 \times 10^{-5}}$	**0.1346**	**0.2572**	$\mathbf{3.85 \times 10^{-10}}$	$\mathbf{9.44 \times 10^{-6}}$	**0.5556**	**0.4972**	$\mathbf{6.37 \times 10^{-10}}$	$\mathbf{1.47 \times 10^{-5}}$	$\mathbf{6.41 \times 10^{2}}$	**1.4923**
LSTM	6.55×10^{-9}	4.69×10^{-5}	0.6089	0.5170	2.38×10^{-9}	1.72×10^{-5}	0.8160	0.6307	2.67×10^{-9}	2.59×10^{-5}	2.75×10^{4}	7.9341
CEEMDAN-BP	2.42×10^{-9}	2.93×10^{-5}	0.2220	0.3013	9.28×10^{-10}	1.27×10^{-5}	1.1358	0.7172	1.52×10^{-9}	2.28×10^{-5}	6.44×10^{3}	4.1960
BP	7.96×10^{-9}	5.67×10^{-5}	1.0587	0.6895	1.98×10^{-9}	2.49×10^{-5}	6.7716	1.8409	4.24×10^{-9}	4.07×10^{-5}	5.96×10^{4}	12.0566
CEEMDAN-ELMAN	2.42×10^{-9}	2.98×10^{-5}	0.2411	0.3202	9.18×10^{-10}	1.21×10^{-5}	0.9015	0.6194	1.19×10^{-9}	1.81×10^{-5}	7.56×10^{2}	1.6656
ELMAN	7.86×10^{-9}	6.54×10^{-5}	1.6260	0.9472	2.47×10^{-9}	3.56×10^{-5}	18.8931	3.0903	5.11×10^{-9}	5.70×10^{-5}	1.14×10^{5}	16.8063
CEEMDAN-SVR	2.45×10^{-9}	2.93×10^{-5}	0.2242	0.3046	9.68×10^{-10}	1.27×10^{-5}	1.0064	0.6772	1.53×10^{-9}	1.97×10^{-5}	2.08×10^{3}	2.5070
SVR	6.36×10^{-9}	4.52×10^{-5}	0.5092	0.4681	1.86×10^{-9}	1.87×10^{-5}	2.9151	1.1392	3.64×10^{-9}	3.77×10^{-5}	7.39×10^{4}	13.1765
CEEMDAN-AR	2.43×10^{-9}	2.97×10^{-5}	0.2151	0.3038	9.85×10^{-10}	1.21×10^{-5}	0.6637	0.5548	1.21×10^{-9}	1.86×10^{-5}	1.45×10^{3}	2.1361
AR	8.79×10^{-9}	7.51×10^{-5}	2.2679	1.1451	2.64×10^{-9}	3.86×10^{-5}	23.4603	3.4402	5.72×10^{-9}	6.32×10^{-5}	1.43×10^{5}	18.8286
HAR	5.69×10^{-9}	5.07×10^{-5}	0.8248	0.6711	1.98×10^{-9}	2.12×10^{-5}	3.9689	1.3910	2.76×10^{-9}	3.26×10^{-5}	5.64×10^{4}	11.4191

Notes: Numbers in bold mean that their corresponding model has the lowest value of loss function among all models.

Table 5
Forecasting evaluation results over different time horizons (MSE).

	CSI 300			S&P 500			STOXX 50		
Models	5 days	21 days	252 days	5 days	21 days	252 days	5 days	21 days	252 days
CEEMDAN-LSTM	2.49×10^{-10}	3.24×10^{-10}	$\mathbf{1.09 \times 10^{-9}}$	$\mathbf{2.31 \times 10^{-11}}$	$\mathbf{5.55 \times 10^{-11}}$	$\mathbf{6.61 \times 10^{-11}}$	$\mathbf{4.09 \times 10^{-9}}$	$\mathbf{2.20 \times 10^{-9}}$	$\mathbf{7.21 \times 10^{-10}}$
LSTM	2.03×10^{-10}	5.02×10^{-10}	4.42×10^{-9}	8.85×10^{-11}	1.83×10^{-10}	1.80×10^{-10}	1.54×10^{-8}	4.44×10^{-9}	3.12×10^{-9}
CEEMDAN-BP	1.34×10^{-10}	2.42×10^{-10}	2.02×10^{-9}	4.91×10^{-11}	7.62×10^{-11}	9.83×10^{-11}	1.05×10^{-8}	3.19×10^{-9}	1.89×10^{-9}
BP	5.30×10^{-10}	8.40×10^{-10}	6.99×10^{-9}	2.78×10^{-10}	4.26×10^{-10}	4.38×10^{-10}	2.86×10^{-8}	7.80×10^{-9}	5.46×10^{-9}
CEEMDAN-ELMAN	1.12×10^{-10}	2.66×10^{-10}	2.03×10^{-9}	5.44×10^{-11}	7.55×10^{-11}	9.28×10^{-11}	6.49×10^{-9}	2.79×10^{-9}	1.50×10^{-9}
ELMAN	2.91×10^{-9}	3.52×10^{-10}	7.25×10^{-9}	8.27×10^{-10}	9.83×10^{-10}	1.06×10^{-9}	2.29×10^{-8}	7.52×10^{-9}	6.04×10^{-9}
CEEMDAN-SVR	$\mathbf{1.06 \times 10^{-10}}$	$\mathbf{2.12 \times 10^{-10}}$	2.03×10^{-9}	5.11×10^{-11}	8.44×10^{-11}	9.80×10^{-11}	1.18×10^{-8}	3.34×10^{-9}	2.01×10^{-9}
SVR	1.27×10^{-10}	3.80×10^{-10}	5.48×10^{-9}	8.88×10^{-11}	2.14×10^{-10}	2.20×10^{-10}	2.63×10^{-8}	7.07×10^{-9}	4.55×10^{-9}
CEEMDAN-AR	1.60×10^{-10}	2.88×10^{-10}	2.04×10^{-9}	6.60×10^{-11}	5.99×10^{-11}	9.16×10^{-11}	6.54×10^{-9}	2.90×10^{-9}	1.52×10^{-9}
AR	5.06×10^{-9}	5.88×10^{-10}	8.40×10^{-9}	1.02×10^{-9}	1.17×10^{-9}	1.29×10^{-10}	2.20×10^{-8}	7.71×10^{-9}	6.60×10^{-9}
HAR	1.03×10^{-9}	1.22×10^{-10}	5.08×10^{-9}	2.15×10^{-10}	2.67×10^{-10}	2.57×10^{-10}	1.82×10^{-8}	5.44×10^{-9}	3.50×10^{-9}

Notes: Numbers in bold mean that their corresponding model has the lowest MSE value among all models.

Table 6
Forecasting evaluation results using the MCS test.

	CSI 300				S&P 500				STOXX 50			
Models	MSE	MAE	HMSE	HMAE	MSE	MAE	HMSE	HMAE	MSE	MAE	HMSE	HMAE
Panel A:T_R												
CEEMDAN-LSTM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LSTM	0.0072	0.0000	0.0065	0.0000	0.0604	0.0019	0.0279	0.0001	0.0215	0.0000	0.0764	0.0764
CEEMDAN-BP	0.0346	0.0012	0.1004	0.0104	0.0604	0.0022	0.0017	0.0000	0.0215	0.0000	0.0766	0.0764
BP	0.0326	0.0000	0.0065	0.0000	0.0604	0.0019	0.0000	0.0000	0.0215	0.0000	0.0764	0.0764
CEEMDAN-ELMAN	0.0346	0.0010	0.0649	0.0004	0.0604	0.0095	0.0015	0.0001	0.0298	0.0004	0.1163	0.0764
ELMAN	0.0072	0.0000	0.0065	0.0000	0.0604	0.0000	0.0000	0.0000	0.0215	0.0000	0.0764	0.0764
CEEMDAN-SVR	0.0346	0.0012	0.0955	0.0078	0.0604	0.0030	0.0015	0.0000	0.0298	0.0004	0.0833	0.0764
SVR	0.0072	0.0000	0.0065	0.0004	0.0604	0.0019	0.0015	0.0000	0.0215	0.0000	0.0764	0.0764
CEEMDAN-AR	0.0326	0.0010	0.1004	0.0078	0.0604	0.0095	0.1787	0.0152	0.0215	0.0000	0.0833	0.0764
AR	0.0072	0.0000	0.0065	0.0000	0.0604	0.0001	0.0000	0.0000	0.0215	0.0000	0.0766	0.0764
HAR	0.0072	0.0000	0.0065	0.0000	0.0604	0.0019	0.0000	0.0000	0.0215	0.0000	0.0764	0.0764
Panel B:T_{SQ}												
CEEMDAN-LSTM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LSTM	0.0076	0.0000	0.0013	0.0000	0.0754	0.0014	0.0374	0.0001	0.0124	0.0000	0.0833	0.0764
CEEMDAN-BP	0.0888	0.0015	0.1401	0.0141	0.1037	0.0019	0.0007	0.0000	0.0219	0.0000	0.0874	0.0764
BP	0.0108	0.0000	0.0001	0.0000	0.0486	0.0000	0.0000	0.0000	0.0124	0.0000	0.0791	0.0764
CEEMDAN-ELMAN	0.0888	0.0008	0.0055	0.0001	0.1037	0.0107	0.0004	0.0000	0.0270	0.0003	0.1163	0.0766
ELMAN	0.0019	0.0000	0.0000	0.0000	0.0163	0.0000	0.0000	0.0000	0.0001	0.0000	0.0814	0.0764
CEEMDAN-SVR	0.0888	0.0015	0.1401	0.0141	0.1037	0.0037	0.0004	0.0000	0.0270	0.0000	0.1008	0.0764
SVR	0.0066	0.0000	0.0013	0.0000	0.0679	0.0001	0.0001	0.0000	0.0079	0.0003	0.0803	0.0764
CEEMDAN-AR	0.0888	0.0013	0.1401	0.0141	0.1037	0.0107	0.1787	0.0152	0.0219	0.0000	0.0940	0.0766
AR	0.0004	0.0000	0.0000	0.0000	0.0055	0.0000	0.0000	0.0000	0.0000	0.0000	0.0846	0.0764
HAR	0.0057	0.0000	0.0000	0.0000	0.0614	0.0000	0.0000	0.0000	0.0073	0.0000	0.0778	0.0764

Notes: The numbers in bold indicate that the corresponding models have best forecasting performance under the MCS criterion. The numbers with p-values larger than 0.25 are underlined.

value larger than 25% can survive the MCS test. The closer p-value is to 1, the better the model's performance is.

For all indices, the empirical results in Table 6 indicate that CEEMDAN-LSTM is still the best model on account that all p-values of CEEMDAN-LSTM under four loss functions equal to 1.

4.3. Robustness check

The heteroscedasticity of RV has been confirmed. RV's characteristics would change at different times, so we adopt a sliding window approach to testify the robustness of empirical results in Section 4.2. The last 90% data of the whole data set are divided into a training set, a validation set, and a test set by 7:1:1. The MCS test results of the robustness check are shown in Table 7. It can be observed that for T_R and T_{SQ}, CEEMDAN-LSTM still performs better than any other models, which confirms the robustness of CEEMDAN-LSTM on different data sets.

5. Conclusions

This paper adopts CEEMDAN to decompose the original RV series and LSTM to forecast the decomposed series, hoping to forecast RV accurately. Then we select AR, HAR, BP, ELMAN, and their hybrid models (HAR excepted) as the comparison. After that, we adopted four loss functions and the MCS test as the performance criteria to testify the superiority of CEEMDAN-LSTM. The test results show that CEEMDAN-LSTM has the best performance in emerging markets and developed

Table 7
Forecasting evaluation results using the MCS test (sliding window).

	CSI 300				S&P 500				STOXX 50			
Models	MSE	MAE	HMSE	HMAE	MSE	MAE	HMSE	HMAE	MSE	MAE	HMSE	HMAE
Panel A:T_R												
CEEMDAN-LSTM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LSTM	0.0185	0.0000	0.0539	0.0000	0.0770	0.0023	0.0003	0.0000	0.0959	0.0042	0.0848	0.0000
CEEMDAN-BP	0.5542	0.1434	0.5913	0.6007	0.0770	0.0235	0.0670	0.0233	0.0959	0.0365	0.3757	0.0990
BP	0.5525	0.0000	0.0539	0.0000	0.0770	0.0235	0.0003	0.0233	0.0959	0.0050	0.0785	0.0000
CEEMDAN-ELMAN	0.5525	0.1434	0.5913	0.8643	0.0928	0.0235	0.8145	0.4657	0.0959	0.0365	0.3757	0.1512
ELMAN	0.0185	0.0000	0.0539	0.0000	0.0770	0.0023	0.0003	0.0000	0.0959	0.0020	0.0345	0.0000
CEEMDAN-SVR	0.5525	0.1323	0.6142	0.8643	0.0770	0.0023	0.0003	0.4657	0.0959	0.0050	0.0785	0.0138
SVR	0.0185	0.0000	0.0539	0.0000	0.0760	0.0235	0.0670	0.0276	0.0959	0.0224	0.0824	0.0000
CEEMDAN-AR	0.5542	0.1323	0.6142	0.9602	0.0928	0.0235	0.8578	0.6822	0.0959	0.0365	0.3757	0.1512
AR	0.0185	0.0000	0.0539	0.0000	0.0760	0.0023	0.0003	0.0233	0.0959	0.0008	0.0345	0.0000
HAR	0.0185	0.0000	0.0539	0.0000	0.0770	0.0023	0.0670	0.0233	0.0959	0.0017	0.0824	0.0000
Panel B:T_{SQ}												
CEEMDAN-LSTM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LSTM	0.0587	0.0001	0.0799	0.0001	0.0915	0.0073	0.0009	0.0000	0.0623	0.0017	0.0799	0.0000
CEEMDAN-BP	0.8147	0.3043	0.0839	0.1052	0.0915	0.0274	0.2019	0.0018	0.1421	0.0306	0.1482	0.0687
BP	0.1466	0.0000	0.0000	0.0000	0.0868	0.0238	0.0009	0.0000	0.1421	0.0291	0.0799	0.0159
CEEMDAN-ELMAN	0.8147	0.3043	0.0839	0.1052	0.1050	0.0274	0.7669	0.2808	0.1421	0.0291	0.1482	0.1324
ELMAN	0.0384	0.0000	0.0000	0.0000	0.0868	0.0049	0.0009	0.0000	0.1421	0.0017	0.0799	0.0000
CEEMDAN-SVR	0.3740	0.1384	0.6279	0.1503	0.0915	0.0073	0.0009	0.0668	0.1421	0.0291	0.0799	0.0687
SVR	0.0507	0.0000	0.0040	0.0000	0.0862	0.0238	0.0406	0.0018	0.1421	0.0291	0.0799	0.0000
CEEMDAN-AR	0.8147	0.3043	0.6279	0.9602	0.1050	0.0274	0.8578	0.6822	0.1421	0.0306	0.1482	0.1161
AR	0.0196	0.0000	0.0000	0.0000	0.0868	0.0044	0.0009	0.0000	0.1421	0.0017	0.0799	0.0000
HAR	0.0359	0.0000	0.0003	0.0000	0.0868	0.0056	0.0043	0.0000	0.1421	0.0017	0.0799	0.0000

Notes: The numbers in bold indicate that the corresponding models have the best forecasting performance under the MCS criterion. The numbers with *p*-values larger than 0.25 are underlined.

markets. Moreover, hybrid models with CEEMDAN outperform their corresponding single models. Furthermore, the results of the robustness check further demonstrate the superiority of CEEMDAN-LSTM on forecasting RV.

CRediT authorship contribution statement

Yu Lin: Data curation, Investigation, Methodology, Resources, Supervision, Writing – original draft, Writing – review & editing. **Zixiao Lin:** Data curation, Investigation, Methodology, Software, Validation, Writing – original draft, Writing – review & editing. **Ying Liao:** Supervision, Validation, Writing – review & editing. **Yizhuo Li:** Supervision, Validation, Writing – review & editing. **Jiali Xu:** Data curation, Software, Resources, Supervision, Writing – review & editing. **Yan Yan:** Supervision, Validation, Writing – review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

All authors appreciate anonymous reviewers and editors for insightful comments and valuable suggestions; this paper's quality has been greatly improved. This paper was supported by the National Natural Science Foundation of China (No. 71771032).

References

Andersen, T. G., & Bollerslev, T. (1998). Answering the skeptics: Yes, standard volatility models do provide accurate forecasts. *International Economic Review, 39*(4), 885–905.
Andersen, T. G., Bollerslev, T., & Diebold, F. X. (2002). Parametric and nonparametric volatility measurement. *SSRN Electronic Journal*, 67–137.
Andersen, T. G., Bollerslev, T., Diebold, F. X., & Labys, P. (1999). *Manuscript, Northwestern University, Duke University and University of Pennsylvania*.
Andersen, T. G., Bollerslev, T., Diebold, F. X., & Labys, P. (2003). Modeling and forecasting realized volatility. *Econometrica, 71*(2), 579–625.
Arnerić, J., Poklepović, T., & Teai, J. W. (2018). Neural network approach in forecasting realized variance using high-frequency data. *Business Systems Research Journal, 9*(2), 18–34.
Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics, 31*, 307–327.
Bucci, A. (2019). Realized Volatility Forecasting with Neural Networks. Working paper.
Cao, J., Li, Z., & Li, J. (2019). Financial time series forecasting model based on CEEMDAN and LSTM. *Physica A: Statistical Mechanics and its Applications, 519*, 127–139.
Chen, W., Ma, F., Wei, Y., & Liu, J. (2020). Forecasting oil price volatility using high-frequency data: New evidence. *International Review of Economics and Finance, 66*, 1–12.
Corsi, F. (2009). A simple approximate long-memory model of realized volatility. *Journal of Financial Econometrics, 7*(2), 174–196.
Duan, Y., Chen, W., Zeng, Q., & Liu, Z. (2018). Leverage effect, economic policy uncertainty and realized volatility with regime switching. *Physica A: Statistical Mechanics and its Applications, 493*(C), 148–154.
Engle, R. F. (1982). Autoregressive conditional heteroscedasticity with estimator of the variance of United Kingdom inflation. *Econometrica, 50*(4), 987–1008.
Hansen, P., Lunde, A., & Nason, J. M. (2011). The model confidence set. *Econometrica, 79*, 453–497.
Hillebrand, E., & Medeiros, M. C. (2010). The benefits of bagging for forecast models of realized volatility. *Econometric Reviews, 29*(5–6), 571–593.
Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735–1780.
Huang, N. E., Shen, Z., Long, S. R., Wu, M. C., Shih, H. H., Zheng, Q., …Liu, H. H. (1998). The empirical mode decomposition and the Hilbert spectrum for non-linear and non-stationary time series analysis. In Proceedings of the royal society of london a: Mathematical, physical and engineering sciences: 454 (pp. 903–995). The Royal Society.
Kat, H. M., & Heynen, R. C. (1994). Volatility prediction: A comparison of the stochastic volatility, GARCH (1, 1) and Egarch (1, 1) models. *Journal of Derivatives, 2*(2), 50–65.
Kim, H. Y., & Won, C. H. (2018). Forecasting the volatility of stock price index: A hybrid model integrating LSTM with multiple GARCH-type models. *Expert Systems with Applications, 103*, 25–37.
Kodama, O., Pichl, L., & Kaizoji, T. (2017). Regime change and trend prediction for Bitcoin time series data. *CBU International Conference Proceedings, 2017*, 5.
Lu, X., Que, D., & Cao, G. (2016). Volatility forecast based on the hybrid artificial neural network and GARCH-type models. *Procedia Computer Science, 91*, 1044–1049.
Luo, J., Ji, Q., Klein, T., Ned, T., & Zhang, D. (2020). On realized volatility of crude oil futures markets: Forecasting with exogenous predictors under structural breaks. *Energy Economics, 89*, Article 104781.

Ma, F., Liu, J., Huang, D., & Chen, W. (2017). Forecasting the oil futures price volatility: A new approach. *Economic Modelling, 64*, 560–566.
Ma, F., Wahab, M. I. M., Huang, D., & Xu, W. (2017). Forecasting the realized volatility of the oil futures market: A regime switching approach. *Energy Economics, 67*, 136–145.
McAleer, M., & Medeiros, M. (2011). Forecasting realized volatility with linear and non-linear univariate models. *Journal of Economic Surveys, 25*(1), 6–18.
Miura, R., Pichl, L., & Kaizoji, T. (2019). *Artificial Neural Networks for Realized Volatility Prediction in Cryptocurrency Time Series* (pp. 165–172). Springer Verlag.
Premanode, B., & Toumazou, C. (2013). Improving prediction of exchange rates using differential EMD. *Expert Systems with Applications, 40*(1), 377–384.
Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research, 15*(1), 1929–1958.
Takahashi, M., Omori, Y., & Watanabe, T. (2009). Estimating stochastic volatility models using daily returns and realized volatility simultaneously. *Computational Statistics & Data Analysis, 53*(6), 2404–2426.
Torres, M. E., Colominas, M. A., Schlotthauer, G., & Flandrin, P. (2011). A complete ensemble empirical mode decomposition with adaptive noise. In *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 4144–4147).
Wei, Y., Wang, Y., & Huang, D. (2010). Forecasting crude oil market volatility: Further evidence using GARCH-class models. *Energy Economics, 32*, 1477–1484.
Wu, Z. H., & Huang, N. E. (2009). Ensemble empirical mode decomposition: A noise assisted data analysis method. *Advances in Adaptive Data Analysis, 1*, 1–41.
Zhang, L., Mykland, P. A., & Ait-Sahalia, Y. (2005). A tale of two time scales: Determining integrated volatility with noisy high frequency data. *Journal of the American Statistical Association, 100*, 1394–1411.
Zhao, Z., Chen, W., Wu, X., Chen, P. C., & Liu, J. (2017). LSTM network: A deep learning approach for short-term traffic forecast. *IET Intelligent Transport Systems, 11*(2), 68–75.