Hindawi
Mathematical Problems in Engineering
Volume 2021, Article ID 8850600, 8 pages
https://doi.org/10.1155/2021/8850600

Research Article

Evolutionary Framework with Bidirectional Long Short-Term Memory Network for Stock Price Prediction

Hongying Zheng,[1] Hongyu Wang,[1,2] and Jianyong Chen [2]

[1]*Sino-German Robotics School, Shenzhen Institute of Information Technology, Shenzhen 518172, China*
[2]*Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ), Shenzhen University, Shenzhen 518060, China*

Correspondence should be addressed to Jianyong Chen; jychen@szu.edu.cn

Received 29 September 2020; Revised 23 August 2021; Accepted 15 September 2021; Published 5 October 2021

Academic Editor: Wei-Chiang Hong

As an important part of the social economy, stock market plays an important role in economic development, and accurate prediction of stock price is important as it can lower the risk of investment decision-making. However, the task of predicting future stock price is very difficult. This difficulty arises from stocks with nonstationary behavior and without any explicit form. In this paper, we propose a novel bidirectional Long Short-Term Memory Network (BiLSTM) framework called evolutionary BiLSTM (EBiLSTM) for the prediction of stock price. In the framework, three independent BiLSTMs correspond to different objective functions and act as mutation individuals, then their respective losses for evolution are calculated, and finally, the optimal objective function is identified by the minimum of loss. Since BiLSTM is effective in the prediction of time series and the evolutionary framework can get an optimal solution for multiple objectives, their combination well adapts to the nonstationary behavior of stock prices. Experiments on several stock market indexes demonstrate that EBiLSTM can achieve better prediction performance than others without the evolutionary operator.

1. Introduction

It is essential for investors to forecast the future price of a stock because the risk of decision-making can be mitigated by appropriately determining its future movement. The topic has attracted many researchers from various academic fields. However, it is a challenging task to predict accurately. In the early stage, most of them used moving average [1], linear regression [2], hidden Markov model (HMM) [3], autoregressive integrated moving average (ARIMA) [4], and prophet model [5] to predict stock prices and their trend.

Currently, neural networks based on deep learning are dominant so far in time series prediction as surveyed, especially Long Short-Term Memory (LSTM) [6]. Recurrent Neural Network (RNN) is also widely used to predict stock prices [7], which applies a decision-making method based on an estimate of the zero-crossing rate to enhance the ability of prediction. Relative insensitivity to gap length is an advantage of LSTM over CNN, RNN, HMM, and other learning methods in numerous applications [8–10]. In some of the initial researches, scholars used only raw financial data for price prediction, which utilized LSTM to predict high and low prices of soybean futures using the data set from the Dalian Commodity Exchange [11]. Later some researchers found that preprocessing of the original data can improve the accuracy of the prediction [12], which proposed a movement trend-based data prediction method to preprocess the trend indicator.

Deep neural networks (DNN) have also been widely applied in stock price prediction to identify trends and patterns. Go and Hong [13] firstly trained DNN by the data of financial time series and then tested and confirmed the predictability of their model. Fluctuation of the stock price was predicted by DNN with 715 novel input features [14]. The performance of their model was also compared with the other models with simple price-based input features. To predict the stock market behavior, the performance of DNN was examined in which high-frequency intraday stock

returns were considered as the input [15]. In the model, the predictability of principal component analysis (PCA), autoencoder, and restricted Boltzmann machine (RBM) was analyzed. The results showed that DNN has good predictability with the information received from the residuals of the autoregressive mode. Moreover, it has been found that financial news may be one of the key factors to produce fluctuations in stock prices. Several sentiment analysis studies have tried to point out the relationship between reaction from investors and news [16], which utilized a novel two-stream gated recurrent unit network to predict the directions of stock prices by using Stock2Vec.

However, due to the complexity of stock data, it is difficult to obtain satisfactory accuracy by only using simple preprocessing or a single LSTM model. By both the complex preprocessing and a hybrid model, the prediction accuracy can be significantly improved [16, 17]. In [17], a large-scale deep learning model was proposed to predict price movements from data of Limit Order Book (LOB) [18]. The architecture utilized CNN to learn the spatial feature of the LOBs and LSTM to remember longer time dependencies series. Framework with multiple LSTMs was also studied extensively to improve the performance of prediction, in which different LSTM can capture different features of data [19, 20].

On the other hand, evolutionary algorithms are inspired by biological evolution mechanisms and simulating evolutionary processes such as reproduction, mutation, genetic recombination, and natural selection, for evolutionary calculation of candidate solutions to optimization problems. Since the evolutionary algorithm is a highly robust and widely applicable global optimization algorithm [21], many scholars have begun to use it to optimize various complex models.

The combination of evolutionary algorithms and neural networks can further improve network performance. In recent years, there have been many achievements in practical applications in multiple fields [22–29]. To minimize human participation in designing deep learning algorithms and automatically discover such configurations, there have been some attempts to optimize deep learning hyperparameters through an evolutionary search [22, 23]. For network optimization, Generative Adversarial Network (GAN) [24] can generate attacked images by one-pixel adversarial perturbation based on differential evolution (DE), that is, black-box attack, which only required a little adversarial information and can fool many types of networks due to the inherent features of DE. Moreover, GAN can also make the generated image more stable through an evolutionary algorithm [25], in which the adversarial game was composed of a population of generators and acted as the discriminator, thereby improving the generative performance. In reinforcement learning, the network topology can also be optimized by combining differential evolution and metaheuristic algorithms [26]. In [27], the transfer learning was used as agents and embedded in the neural network to determine which parts of the network can be reused for a new task. During learning, a tournament selection genetic algorithm was used to select pathways through the neural networks. LSTM combined with evolutionary algorithms has been reported in the prediction of time series [28, 29]. In [28], the gradient descent method in LSTM was combined with the particle swarm optimization (PSO) algorithm to update the weights of network. In [29], an evolutionary attention-based LSTM was proposed for multivariate time series prediction, which can refrain from being trapped into partial optimization like traditional gradient-based methods.

In this paper, EBiLSTM is proposed for stock price prediction, which takes the BiLSTM training procedure as an evolutionary problem. Specifically, the training process of each BiLSTM has its adaptive loss functions and is independent. A population of BiLSTMs evolves corresponding to the training process of multi-BiLSTMs. The BiLSTM is trained for predicting the stock price of the next day during each training (or evolutionary) iteration.

In summary, contributions in this paper are listed as follows:

(i) A framework named EBiLSTM is proposed which integrates BiLSTM and evolutionary algorithm to effectively predict stock price. As far as we know, it is the first report on the approach.

(ii) An evolution strategy is proposed which uses multiobjective functions (square loss, abs loss, and Huber loss) to optimize BiLSTM.

(iii) Performances are evaluated with several stock market indexes and the results demonstrate that the proposed EBiLSTM can get more accuracy of prediction than others.

The rest of this paper is organized as follows: in Section 2, EBiLSTM together with its training process is proposed. Section 3 provides the experimental validation of the method. Finally, the conclusion is presented in Section 4.

2. Method

2.1. Framework. In contrast to conventional BiLSTM which utilizes a single BiLSTM to train the stock data of the real world, an evolutionary algorithm is used that evolves a population of BiLSTM (s), that is, {BiLSTM}, in the training process. In this population, each individual stands for a possible solution in the weights space of the BiLSTM. During the evolutionary process, mutation operations (different objective functions are chosen dynamically) are used to generate different offspring individuals (the weights of different BiLSTM). As shown in Figure 1, there are three substages in each step of evolution:

(i) Mutation: Given an individual BiLSTM_θ in the population, the variation operators are used to produce its offspring $\{\text{BiLSTM}_{\theta 1}, \text{BiLSTM}_{\theta 2}, \ldots\}$. Each individual creates some copies and mutations are used to modify each of them. The modified copies are taken as children.

(ii) Evaluation: BiLSTM training process is taken as fitness function $F(\cdot)$ which is used to evaluate the

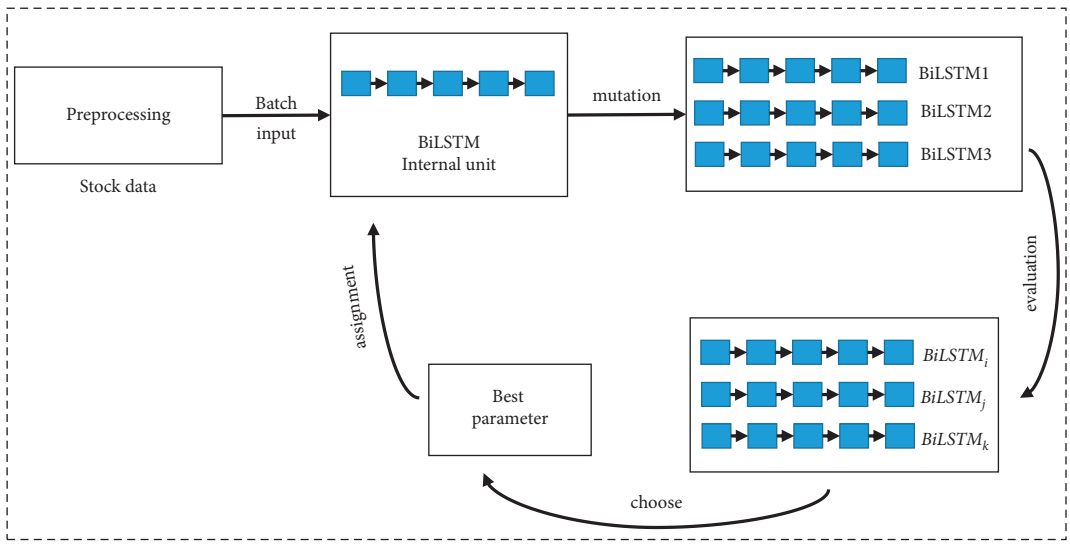

Figure 1: Framework of EBiLSTM.

performance of each child. It can be represented by fitness score.

(iii) Selection: According to the fitness score, all children are sorted from high to low. Some of the lowest ones, that is, the worst ones, are deleted. The remaining children are kept and act as parents to be further evolved at the next iteration.

After each evolutionary circle, the BiLSTM is updated to predict the price of the next day; that is,

$$L_{\text{BiLSTM}} = F_{\text{BiLSTM}}(\text{square, abs or huber}). \quad (1)$$

Here, we take a simple example to show the process of evolution. Data of stock A is split into various batches to train BiLSTM. The first batch, that is, data_1, is input to BiLSTM. According to the difference between the outputs and the corresponding labels, various losses, such as square, abs, and Huber, are calculated. Among them, the least one is selected and the weights of BiLSTM are updated. Then, the second batch, that is, data_2, is input to BiLSTM. The training process continues until the loss becomes small enough. Thus, the adaptive losses from the evolution of the BiLSTM population can produce optimal solutions.

2.2. Mutation. Sexual reproduction with different mutations is employed to produce the next BiLSTM's individuals (i.e., children). Specifically, these mutation operations are taken as different training objectives, which try to narrow the distance between the predicted value and the real stock price. In this section, the mutation is presented in detail. To analyze the properties of these mutations, we assume that the optimal BiLSTM model can be got from each evolutionary circle.

(1) Abs mutation (L1 loss): The abs mutation represents the abs objective function in the original BiLSTM:

$$M_{\text{abs}} = |y - f(x)|. \quad (2)$$

The abs aims to minimize the absolute value between the prediction value and the real close price (label). If there are outliers in the training set which may corrupt the training process, it is necessary to use L1 loss. However, L1 loss has a serious problem. Its gradient is kept the same throughout. When L1 loss is small, the gradient will be large which may impede learning.

(2) Square mutation (L2 loss): The square mutation represents the square objective function in the original BiLSTM:

$$M_{\text{square}} = (y - f(x))^2. \quad (3)$$

Gradients of the square mutation can be used for BiLSTM training. When L2 loss approaches zero, it means that the prediction accuracy of BiLSTM is very high (i.e., $L_{\text{BiLSTM}} \longrightarrow 0$). While when L2 loss is close to infinity, it means that the training of BiLSTM is not effective. Because L2 loss is square of the error ($y - f(x) = e$), the error (e) increases a lot when $e > 1$. Once there is an outlier in our data, e may be high and e^2 may be $\gg |e|$. Due to the outliers, the weights of a model will be affected more seriously by L2 loss than by L1 loss.

(3) Huber mutation: the Huber mutation represents the Huber objective function in the original BiLSTM:

$$M_{\text{huber}} = \begin{cases} \frac{1}{2}[y - f(x)]^2 & |y - f(x)| \le \delta, \\ \delta|y - f(x)| - \frac{1}{2\delta^2} & |y - f(x)| > \delta. \end{cases} \quad (4)$$

2.3. Evaluation. For evolutionary algorithm, fitness function (i.e., evaluation operation) is used to measure the quality of an individual. In this paper, the loss function is taken as the

fitness function which focuses on minimal loss. As shown in equation (1), the smaller the loss value is, the better the fitness is.

Note that BiLSTM is constantly updated to get the optimal solution in the training process. If a BiLSTM has a relatively high fitness value, its prediction result can get better performance.

2.4. Selection. In an evolutionary algorithm, the selection is the counterpart of the mutation operators. In the proposed EBiLSTM, a simple yet useful survivor selection strategy is used to determine the next generation based on the fitness score of existing individuals.

Particularly, the BiLSTMs (i.e., population) are optimized in a dynamic procedure. Thus, the fitness function is changeable and we can evaluate the fitness score of each BiLSTM from the corresponding training process in the same evolutionary generation. It indicates that we cannot compare the fitness scores evaluated in different generations with each other. Moreover, because the mutation operators of the proposed EBiLSTM actually represent selection from different BiLSTM training objectives, the desired offspring represents the effective training strategies. Taking into account both the fitness function and the mutation operators, the selection mechanism of EBiLSTM is taken as the comma selection, i.e., (μ, λ)-selection. Specifically, after the current offspring population $\{\text{BiLSTM}_i\}_{i=1}^{\lambda}$ is sorted according to their fitness scores F_i, the μ – best individuals are selected as population of the next generation.

2.5. Data Preprocessing. Stock data have variables with multiple dimensions, such as opening price, closing price, the highest price, the lowest price, and trading rate. Among these prices, the closing price is always the most concern by investors. Therefore, we use the closing price as the input variable. Our experiments show that the results are basically the same when other prices are used as input variables.

As shown in Figure 2, the overall data is divided into a training set and a test set, respectively. A rolling window is used to segment data. We use the way of $N+1$ (i.e., the closing price of the previous N days is used to predict the closing price of the next day) to train EBiLSTM continuously. After the train finishes, the data of the last N days in the training set are used as input data to predict the first day of the test set. Then the rolling window of input data is shifted forward one day and the last day of the input data becomes vacant. In this case, the first day of the test data is used as the last day of the input data to predict the second day of the test data and so forth. Once the last day of the test set is predicted, the test process is over.

To reduce the noise of stock data and benefit the detection of stock price pattern, it is necessary to smooth and normalize the stock price data since every stock may have its specific domain and scale. Data normalization is defined as adjusting values measured on different scales to a uniform scale [30], while data smoothing is defined as transforming stock prices into variations of daily change. The data smoothing is shown in the following equation:

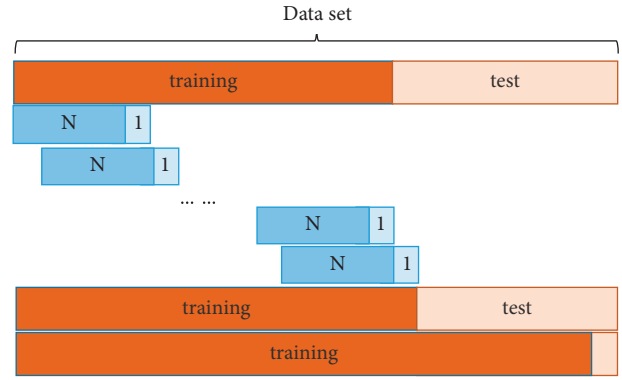

Figure 2: Schematic diagram of the rolling window.

$$x_{(s,t)} = \begin{cases} 0, & t = 1, \\ \dfrac{x_t - x_{t-1}}{x_{t-1}}, & t > 1, \end{cases} \tag{5}$$

where $x_{(s,t)}$ denotes the result of data smoothing at the t^{th} day. Here, when $t = 1$, we set $x_{(s,t)} = 0$.

The learning of BiLSTM is in fact to get stock patterns which can be magnified by "min–max" normalization of the data set. The "min–max" normalization method is shown as follows:

$$x_{(n,t)} = \frac{x_{(s,t)} - x_{(s,\min)}}{x_{(s,\max)} - x_{(s,\min)}}, \tag{6}$$

where $x_{(n,t)}$ denotes the data after normalization, $x_{(s,t)}$ is original data, $x_{(s,\min)}$ is the minimum among the data set, and $x_{(s,\max)}$ is the maximum.

Accordingly, denormalization and desmoothness are required at the end of the prediction process to get the original price, which are given by

$$\hat{x}_{(s,t)} = \hat{x}_{(n,t)}\left[x_{(s,\max)} - x_{(s,\min)}\right] + x_{(s,\min)}, \tag{7}$$

$$\hat{x}_t = \hat{x}_{(s,t)} x_{t-1} + x_{t-1}, \tag{8}$$

where $\hat{x}_{(n,t)}$ denotes the predicted data, $\hat{x}_{(s,t)}$ denotes the predicted data after denormalization, and $\hat{x}_t$ denotes the predicted data after both denormalization and desmoothness.

2.6. BiLSTM. As a variant of LSTM [6], BiLSTM can capture context information more comprehensively and the correlations between contexts. Two LSTM networks, one is with a forward direction and the other is with a backward direction, are connected to the same output layer. Both of them are trained with the same sequence of data. There are three gates, that is, input gate, forget gate, and output gate, in a unit of LSTM. Equations (9)–(14) show the calculation processes:

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

$$f_t = \sigma\left(w_f\left[h_{t-1}, x_t\right] + b_f\right), \tag{10}$$

Input: population size $P = N$, the number of mutations n_m, the batch size m, batch data D, and initial weight ω_0,
output: close price of the next day

(1) $\omega = \omega_0$
(2) Initializes model parameter ω_0:
(3) **for** $i = 1$ **to** $m/(Nn_m)$
(4) $param \longleftarrow \omega$ save model parameters
(5) **for** $j = 1$ **to** N
(6) **for** $k = 1$ **to** n_m
(7) $M(param)$ assign parameters to the model
(8) get a batch D as input x_i of EBiLSTM;
(9) **switch(k)**
(10) **case1:** $loss_{square}, param_{square} \longleftarrow M(x_i, square, param)$
(11) **case2:** $loss_{abs}, param_{abs} \longleftarrow M(x_i, abs, param)$
(12) **case3:** $loss_{huber}, param_{huber} \longleftarrow M(x_i, huber, param)$
(13) **end switch**
(14) **if** $k = n_m$
(15) $loss_{\min} \longleftarrow \min(loss_{square}, loss_{abs}, loss_{huber})$
(16) $param_{new} \longleftarrow (loss_{\min}, param_{square}, param_{abs}, param_{huber})$
(17) $\omega \longleftarrow param_{new}$
(18) **end for**
(19) **end for**
(20) **end for**

Algorithm 1:

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

$$\tilde{c}_t = \tanh(w_c[h_{t-1}, x_t] + b_c), \tag{12}$$

$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t, \tag{13}$$

$$h_t = o_t * \tanh(c_t). \tag{14}$$

Here, w_i, w_f, and w_o are the weights of LSTM and b_i, b_f, and b_o are the biases. i_t is input gate, f_t is forget gate, and o_t is output gate. The input vector is x_t and the output vector is h_t. c_t is the cell state and $t * \tilde{c}_t$ means the candidate of the cell state. For the forward LSTM, it can be presented as $\overrightarrow{h}_t = \overrightarrow{\text{LSTM}}(x_t, \overrightarrow{h}_{t-1})$. Accordingly, the backward LSTM is with $\overleftarrow{h}_t = \overleftarrow{\text{LSTM}}(x_t, \overleftarrow{h}_{t+1})$. Both $\overrightarrow{h}$ and $\overleftarrow{h}$ are the output of BiLSTM at time t,

$$h_t = \left[\overrightarrow{h}_t; \overleftarrow{h}_t\right]. \tag{15}$$

2.7. EBiLSTM. The complete process of EBiLSTM can be shown in Algorithm 1. At each evolutionary circle, BiLSTMs are updated with different mutations (or objectives). Among children of the next generation, only well-performing ones will survive and participate in the next rotation of training, following the principle of "survival of the fittest." Unlike a single BiLSTM with a fixed and static training objective, EBiLSTM integrates the advantages of different training objectives and selects the best solution. Therefore, during training, EBiLSTM can not only largely suppress the limitations (local optimal, etc.) of individual training objectives but also harness their advantages to find a better solution.

3. Experiment

3.1. Implementation Details. In order to evaluate the proposed EBiLSTM, experiments on several stock prediction tasks are run and their prediction results are presented in this section. Comparing with previous BiLSTM models, we show that the proposed EBiLSTM can achieve better stock prediction. Configurations from Table 1 are used in all the following experiments.

We evaluate EBiLSTM with three stock market indexes, as shown in Table 2. Every index includes data of 4750 days which are large enough to train EBiLSTM effectively.

3.2. Evaluation Metrics. To evaluate the proposed EBiLSTM, we use Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), and Mean Square Error (MSE) as quantitative metrics. They are shown as follows:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|\hat{x}_i - x_i\right|, \tag{16}$$

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{\hat{x}_i - x_i}{x_i}\right| \times 100, \tag{17}$$

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{x}_i - x_i\right)^2. \tag{18}$$

For MAE, the error is calculated as an average of absolute differences between the target values and the predictions. It is a linear score and always nonnegative which means that all the individual differences are weighted equally on average [31]. The closer to 0 its value is, the higher the accuracy is. MAPE is measured by calculating the absolute error in each

Table 1: The structure of EBiLSTM and hyperparameter.

Input: real closing price (20 days)
[Layer1] internal cells (32) (BiLSTM | LSTM | GRU | RNN)
Output: predicted price in the next day
Hyperparameter: lr = 0.00001, epoch = 150, and timestep = 21
Survived parents number $\mu = 1$

Table 2: Data period of stock market indexes.

Index	Date
Shanghai Securities Composite Index	From Jan 2, 2001, to Jan 23, 2020
A-Share Index	From Jan 2, 2001, to Jan 23, 2020
Standard & Poor's 500 index	From Sep 8, 2000, to Sep 6, 2019

period, dividing this by actual value for that period, and finding the average of absolute percentage errors. MSE is basically based on the average squared error of our predictions. For each point, it calculates the square difference between the predictions and the target and then averages those values. The higher this value is, the worse the model is.

3.3. Effectiveness. To evaluate the effectiveness of the EBiLSTM, it is compared with BiLSTMs under different loss functions, that is, BiLSTM-square, BiLSTM-abs, and BiLSTM-Huber. After training, closing prices of 50 days are predicted. For each model, simulations are taken 5 times independently and their average results of metrics are shown in Tables 3–5. Three stock market indexes shown in Table 1 are used in simulations, respectively. From these tables, it is evident that EBiLSTM achieves the best performance among the four models. The three models with a single objective function always get worse results because their objective functions cannot keep optimal during training. It may be easier for them to be suffered from local minima of parameters.

3.4. Training Stability. To further examine the performance of prediction for different length of days, EBiLSTM as well as BiLSTM-square with a single objective is simulated with different days of prediction. The results with Shanghai Securities Composite Index are shown in Table 6. In this table, EBiLSTM can always get the best performance at different days of prediction.

3.5. Generality. The architecture of EBiLSTM is general which can integrate different deep learning algorithms and keeps good performance. To demonstrate the generality, LSTM, GRU, and RNN are used to replace BiLSTM in the architecture, which are named ELSTM, EGRU, and ERNN, respectively. Simulation results with Shanghai Securities Composite Index are shown in Tables 7–9. In Table 7, ELSTM is compared with its corresponding algorithms with a single objective, that is, LSTM-square, LSTM-abs, and

Table 3: Standard & Poor's 500.

Metrics	EBiLSTM	BiLSTM-square	BiLSTM-abs	BiLSTM-Huber
MSE	**945**	1253	1107	1191
MAPE	**0.82**	0.91	0.89	0.90
MAE	**23.98**	26.63	25.88	26.20

Table 4: Shanghai Securities Composite Index.

Metrics	EBiLSTM	BiLSTM-square	BiLSTM-abs	BiLSTM-Huber
MSE	**608**	804	713	765
MAPE	**0.61**	0.74	0.68	0.71
MAE	**18.28**	22.09	20.42	21.22

Table 5: A-Share Index.

Metrics	EBiLSTM	BiLSTM-square	BiLSTM-abs	BiLSTM-Huber
MSE	**721**	899	785	901
MAPE	**0.66**	0.73	0.68	0.73
MAE	**20.78**	22.95	21.33	22.79

Table 6: Shanghai Securities Composite Index.

Days	Algorithms	MSE	MAPE	MAE
50	EBiLSTM	**608**	**0.61**	**18.28**
	BiLSTM-square	804	0.74	22.09
100	EBiLSTM	**534**	**0.62**	**18.44**
	BiLSTM-square	701	0.71	21.06
150	EBiLSTM	**637**	**0.66**	**19.57**
	BiLSTM-square	830	0.78	22.94
200	EBiLSTM	**951**	**0.75**	**22.26**
	BiLSTM-square	1231	0.88	26.00

Table 7: Shanghai Securities Composite Index.

Metrics	ELSTM	LSTM-square [34]	LSTM-abs	LSTM-Huber
MSE	**653**	706	649	681
MAPE	**0.65**	0.68	0.65	0.67
MAE	**19.53**	20.38	19.44	20.16

Table 8: Shanghai Securities Composite Index.

Metrics	EGRU	GRU-square	GRU-abs	GRU-Huber
MSE	**584**	844	812	801
MAPE	**0.63**	0.75	0.74	0.73
MAE	**18.95**	22.60	22.11	21.97

Table 9: Shanghai Securities Composite Index.

Metrics	ERNN	RNN -square	RNN -abs	RNN-Huber
MSE	**574**	979	1044	989
MAPE	**0.64**	0.83	0.85	0.83
MAE	**19.17**	25.00	25.49	24.92

LSTM-Huber. The results show that ELSTM is the best among them. Similar results can be got for EGRU and ERNN which are shown in Tables 8 and 9, respectively.

4. Conclusion

Stock market exchanges have become popular, encouraging researchers to find predictions using new technologies or methods. Proper predictive techniques can help investors get higher profits from the stock market. However, it is difficult to improve the prediction only by using neural networks because gradient descent in the training process is easy to fall into local optimal. To overcome it, we propose an evolutionary framework (EBiLSTM) for stock prediction. In the framework, we propose an evolutionary algorithm to evolve a population of BiLSTM. In contrast to conventional BiLSTM, the evolution of EBiLSTM selects three different BiLSTM objective functions as mutated individuals, then calculates their respective losses for evaluation, and finally selects the optimal objective function through the minimum loss. Experiments show that EBiLSTM can improve the training stability of BiLSTM and achieves more accuracy of prediction than others in various stock market indexes. For further investigation, there are still some promising directions in the future. For example, the BiLSTM with attention mechanism might have more potential to get better performance. In forecast-based trading, it is interesting to design a portfolio allocation to improve the performance of BiLSTM.

Data Availability

The data are available at https://tushare.pro/.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was partially supported by the Natural Science Foundation of Guangdong Province (S2018A030313420), Science and Technology Plan Project of Longgang District (KJD2020D004), and Science and Technology Plan Project of Shenzhen Institute of Technology Information (SZIIT2020PT094).

References

[1] S. Lauren and S. D. Harlili, "Stock trend prediction using simple moving average supported by news classification," in *Proceedings of the 2014 International Conference of Advanced Informatics: Concept, Theory and Application (ICAICTA)*, Bandung, Indonesia, August 2014.

[2] A. Izzah, Y. A. Sari, R. Widyastuti, and T. A. Cinderatama, "Mobile app for stock prediction using improved multiple linear regression," in *Proceedings of the 2017 International Conference on Sustainable Information Engineering and Technology (SIET)*, Batu, Java, Indonesia, November 2017.

[3] P. Somani, S. Talele, and S. Sawant, "Stock market prediction using hidden markov model," in *Proceedings of the 2014 IEEE 7th Joint International Information Technology and Artificial Intelligence Conference*, Chongqing, China, December 2014.

[4] L. Xiong and Y. Lu, "Hybrid ARIMA-BPNN model for time series prediction of the Chinese stock market," in *Proceedings of the 2017 3rd International Conference on Information Management (ICIM)*, Chengdu, China, April 2017.

[5] B. L. Pooja, S. Kanakaraddi, and M. M. Raikar, "Sentiment based stock market prediction," in *Proceedings of the 2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS)*, Belgaum, India, December 2018.

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

[7] Y. F. Lin, Y. L. Ueng, W. H. Chung, and T. M. Huang, "Stock price range forecast via a recurrent neural network based on the zero-crossing rate approach," in *Proceedings of the IEEE Conference on Computational Intelligence for Financial Engineering & Economics (CIFEr)*, Shenzhen, China, May 2019.

[8] Y. Xing, C. Lv, and D. Cao, "Personalized vehicle trajectory prediction based on joint time-series modeling for connected vehicles," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 2, pp. 1341–1352, 2020.

[9] M. Hwang, B. Thananjeyan, and S. Paradis, "Efficiently calibrating cable-driven surgical robots with RGBD sensing, temporal windowing, and linear and recurrent neural network compensation," 2020, https://arxiv.org/abs/2003.08520v4.

[10] M. Ravanelli, P. Brakel, M. Omologo, and Y. Bengio, "Light gated recurrent units for speech recognition," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 2, pp. 92–102, 2018.

[11] C. Wang and Q. Gao, "High and low prices prediction of soybean futures with LSTM neural network," in *Proceedings of the IEEE International Conference on Software Engineering and Service Science (ICSESS)*, Beijing, China, November 2018.

[12] Y. Liu and X. Liu, "A trend-based stock index forecasting model with gated recurrent neural network," in *Proceedings of the IEEE International Conference on Progress in Informatics and Computing (PIC)*, Suzhou, China, December 2018.

[13] Y. H. Go and J. K. Hong, "Prediction of stock value using pattern matching algorithm based on deep learning," *International Journal of Recent Technology and Engineering*, vol. 8, no. 2, pp. 31–35, 2019.

[14] Y. Song, J. W. Lee, and J. Lee, "A study on novel filtering and relationship between input-features and target-vectors in a deep learning model for stock price prediction," *Applied Intelligence*, vol. 49, no. 3, pp. 897–911, 2019.

[15] E. Chong, C. Han, and F. C. Park, "Deep learning networks for stock market analysis and prediction: methodology, data representations, and case studies," *Expert Systems with Applications*, vol. 83, pp. 187–205, 2017.

[16] D. Lien Minh, A. Sadeghi-Niaraki, H. D. Huy, K. Min, and H. Moon, "Deep learning approach for short-term stock trends prediction based on two-stream gated recurrent unit network," *IEEE Access*, vol. 6, pp. 55392–55404, 2018.

[17] Z. Zhang, S. Zohren, and S. Roberts, "DeepLOB: deep convolutional neural networks for limit order books," *IEEE Transactions on Signal Processing*, vol. 67, no. 11, pp. 3001–3012, 2019.

[18] C. A. Parlour and D. J. Seppi, "Limit order markets: a survey," *Handbook of Financial Intermediation and Banking*, Elsevier, vol. 5, pp. 63–96, Amsterdam, Netherlands, 2008.

[19] X. Shao, D. Ma, Y. Liu, and Q. Yin, "Short-term forecast of stock price of multi-branch LSTM based on K-means," in *Proceedings of the 2017 4th International Conference on*

Systems and Informatics (ICSAI), Hangzhou, China, November 2017.

[20] O. Orojo, J. Tepper, T. M. McGinnity, and M. Mahmud, "A multi-recurrent network for crude oil price prediction," in *Proceedings of the 2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, Xiamen, China, December 2019.

[21] F. G. Mohammadi, M. H Amini, and H. R. Arabnia, "Evolutionary computation, optimization and learning algorithms for data science," 2019, https://arxiv.org/abs/1908.08006v1.

[22] R. Miikkulainen, J. Liang, and E. Meyerson, "Evolving deep neural networks," 2017, https://arxiv.org/abs/1703.00548v2.

[23] S. R. Young, D. C. Rose, and T. P. Karnowski, "Optimizing deep learning hyper-parameters through an evolutionary algorithm," in *Proceedings of the Workshop on Machine Learning in High-Performance Computing Environments*, Austin, Texas, November 2015.

[24] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 5, pp. 828–841, 2019.

[25] C. Wang, C. Xu, X. Yao, and D. Tao, "Evolutionary generative adversarial networks," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 6, pp. 921–934, 2019.

[26] A. Banerjee, D. Ghosh, and S. Das, "Evolving network topology in policy gradient reinforcement learning algorithms," in *Proceedings of the 2019 Second International Conference on Advanced Computational and Communication Paradigms (ICACCP)*, Gangtok, India, February 2019.

[27] C. Fernando, D. Banarse, and C. Blundell, "PathNet: evolution channels gradient descent in super neural networks," 2017, https://arxiv.org/abs/1701.08734.

[28] Y. Hu, X. Sun, X. Nie, Y. Li, and L. Liu, "An enhanced LSTM for trend following of time series," *IEEE Access*, vol. 7, pp. 34020–34030, 2019.

[29] Y. R. Li, Z. F. Zhu, and D. Q. Kong, "EA-LSTM: evolutionary attention-based LSTM for time series prediction," 2018, https://arxiv.org/abs/1811.03760.

[30] R. Hafezi, J. Shahrabi, and E. Hadavandi, "A bat-neural network multi-agent system (BNNMAS) for stock price prediction: case study of DAX stock price," *Applied Soft Computing*, vol. 29, pp. 196–210, 2015.

[31] T. Chai and R. R. Draxler, "Root mean square error (RMSE) or mean absolute error (MAE)? —arguments against avoiding RMSE in the literature," *Geoscientific Model Development*, vol. 7, no. 3, pp. 1247–1250, 2014.