IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 28, 2020, accepted January 21, 2021, date of publication February 3, 2021, date of current version February 10, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3056713

A Novel Improved Particle Swarm Optimization With Long-Short Term Memory Hybrid Model for Stock Indices Forecast

YI JI [1,2,3], ALAN WEE-CHUNG LIEW [2], (Senior Member, IEEE), AND LIXIA YANG [1], (Member, IEEE)

[1]School of Computer Science and Communication Engineering, Jiangsu University, Zhenjiang, Jiangsu 212013, China
[2]School of Information and Communication Technology, Griffith University, Gold Coast, QLD 4222, Australia
[3]School of Electrical and Information Engineering, Jiangsu University, Zhenjiang, Jiangsu 212013, China

Corresponding authors: Yi Ji (jiyi@ujs.edu.cn) and Alan Wee-Chung Liew (a.liew@griffith.edu.au)

This work was supported in part by the National Natural Science Foundation of China under Grant 41874174, and in part by the Postgraduate Research and Practice Innovation Program of Jiangsu Province under Grant KYCX19_1615.

ABSTRACT Stock market volatility has a significant impact on many economic and financial activities in the world. Forecasting stock price movement plays an important role in setting an investment strategy or determining the right timing for trading. However, stock price movements are noisy, nonlinear, and chaotic. It is difficult to forecast stock trends for improving return on investment. Here, we proposed a novel improved particle swarm optimization (IPSO) and long-short term memory (LSTM) hybrid model for stock price forecasting. An adaptive mutation factor was used as a parameter for model optimization to avoid premature convergence to a local optimum. Furthermore, we presented a nonlinear approach to improve the inertia weight of the particle swarm optimization and then used the IPSO model to optimize the hyperparameters of LSTM. The experimental results showed that the proposed model outperformed the other related baseline models: support-vector regression, LSTM and PSO-LSTM on the Australian stock market index. These results indicated the proposed model possesses high reliability and good forecasting capability.

INDEX TERMS Long-short term memory network, time series forecasting, stock indices forecasting, hybrid feature selection, deep learning.

I. INTRODUCTION

The stock market index is a reference indicator compiled by the stock exchange or financial service institutions to show movement in the stock market. It is an important financial indicator for investors, as the stock index affects the sentiment of investors regarding whether to buy, hold, or sell their stocks. Due to the importance of the stock market to the economy, researchers and investors always pay attention to forecast stock market trends.

During the past decades, the development of machine learning and deep learning methodologies for forecasting stock price and trends, such as artificial neural network (ANN), support vector machine (SVM), and long short-term memory (LSTM) have gained popularity in the scientific and commercial communities. These methodologies can provide useful findings and insight about market trend, and have caught the attention of researchers.

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

Previous studies have demonstrated that ANN and SVM are important models for financial time series forecasting. Tay and Cao applied SVM to forecast future contract prices that were selected from the Chicago Mercantile Exchange (CME) datasets in 2001 [1]. Two years later, Kim utilized SVM to forecast the trends of daily stock price in the Korea Composite Stock Price Index (KOSPI) [2]. Huang *et al.* used SVM to forecast the weekly movement direction of the Nikkei 225 index [3]. Hossain and Nasser predicted changes in the Nikkei 225 and S&P 500 stock indices by the ARMA-GARCH and SVM models [4]. Sheta *et al.* explored ANN, traditional multiple linear regression (MLR), and SVM to establish prediction models for the S&P 500 stock index [5]. Hui and Zhang proposed a novel kernel of support vector regression for forecasting high-frequency equity returns in 2016 [6]. Chi-Yuan Yeh *et al.* applied a multiple-kernel support vector regression approach to forecasting stock market price [7]. Hung carried out a fuzzy support vector regression model for forecasting stock market volatility [8]. Two hybrid models named GA-SVR-GM and

GABPNN-GM are proposed for improving the prediction accuracy of the Composite Index in Shanghai and Shenzhen Stock Exchanges [9].

The stock market is known for its high volatility and uncertainty. Thus, traditional approaches are difficult to achieve accurate prediction. However, in recent years, efforts have been made to forecast stock prices by LSTM network that is good at processing nonlinear financial time series. Chen *et al.* forecasted Chinese stock returns by LSTM [10]. Nelson *et al.* implemented LSTM to predict future stock movements by stock price and technical analysis indicators [11]. Bao *et al.* used a three-stage process to predict six market index futures [12]. First, they used a wavelet transformation to reduce the dimensionality of stock data. Second, they reproduced these data by a stacked autoencoder. Finally, they applied LSTM to predict stock price. They confirmed that the prediction capability of the proposed model was better than the other models, such as RNN, LSTM, and wavelet-LSTM models. Fischer and Krauss in 2018 [13] deployed LSTM networks to forecast the directional movement of constituent stocks of the S&P 500 from 1992 to 2015. They compared the simulation result with memory-free classifiers, such as random forest (RF), deep neural network (DNN), and logistic regression (LR) and found that LSTM significantly outperformed the other comparative models.

LSTM network has memory cells and feedback connections. Many studies have found that LSTM network can obtain valuable features from low volume data [10]–[13]. Furthermore, LSTM is a method especially suitable for processing time series. Nevertheless, the parameters of the LSTM network are usually determined by user's experience. The choice of these parameters will affect the model's forecasting capability. In order to address the above problem, various meta-heuristics optimization algorithms such as simulated annealing, genetic algorithm, particle swarm optimization, and so on, have been proposed for parameter optimization. Lu [14] constructed a hybrid stock index prediction model by utilizing particle swarm optimization (PSO) to optimize the parameters of the SVR model. Shen *et al.* [15] used the artificial fish swarm algorithm (AFSA) to optimize a radial basis function neural network to forecast the Shanghai Security Exchange index. Xia *et al.* [16] proposed an expanded PSO (XPSO) with forgetting capability to improve the convergence rate of the algorithm. Other researchers applied improved optimization algorithms to numerical computing, feature selection, and convergence improvement [17]–[22].

The contributions of this paper are as follows:

(1) To effectively resolve premature convergence issues on the PSO algorithm, we propose novel inertia weight and adaptive mutation factor to greatly improve the particles global search capability and avoid the particles easily fall into the local optimum. (2) The LSTM network hyperparameters are updated adaptively by the IPSO model. Using the optimal LSTM network, we can perform better forecast performance. (3) A suitable look-back period is very important for the LSTM forecast, which can affect the forecasting results directly. The suitable values of look-back period are confirmed properly by analyzing experimental results. The rest of the paper is organized as follows. The theories of SVR, LSTM, PSO, PSO-LSTM are briefly explained in Section 2. In Section 3, we improve the PSO algorithm and determine the hyperparameters of LSTM with IPSO. In Section 4, we compare the forecasting results of LSTM optimized by IPSO, PSO, and the other algorithms, as well as forecasting results of SVR and LSTM. Furthermore, the robustness of IPSO-LSTM is verified by four well-known stock indices. Finally, the conclusion and suggestions for further study are given in Section 5.

II. PRELIMINARIES

A. SUPPORT VECTOR MACHINES FOR REGRESSION

Support Vector Machines is a classic supervised learning technique proposed by V. N. Vapnik in 1995 [23]. A version of the SVM for regression, called support-vector regression (SVR), was developed in 1996 [24]. By using kernels, it is possible to regress a non-linear function for solving nonlinear problems by SVR. The SVR method can be expressed as follows:

Given the training dataset of n elements,

$$\{(x_1, y_1), (x_2, y_2), (x_3, y_3), \cdots, (x_n, y_n)\},$$

where $x_i \in R^n$ represents the i_{th} input data and $y_i \in R^n$ represents the i_{th} output data, $i = 1, 2, 3, \cdots, n$.

In SVR, the training data x_i is mapped to the high n-dimensional feature space that formulates an optimized hyperplane that also represents the non-linear relationship between input and output data. The function is expressed as:

$$f(x) = w^T \cdot \phi(x) + b \tag{1}$$

where w represents the weight vector; $\phi(x)$ represents a kernel function; b represents the bias; x represents the input vector.

The SVR model is designed to minimize the followings:

$$Min : \tfrac{1}{2} w^T w + C \textstyle\sum_i \left(\xi_i + \xi_i^*\right) \tag{2}$$

$$Subject\ to \left\{ \begin{array}{c} y_i - \left(w^T \cdot \phi(x)\right) - b \le \varepsilon + \xi_i \\ w^T \cdot \phi(x) + b - y_i \le \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0, i = 1, 2, \cdots, m \end{array} \right\} \tag{3}$$

where C is the penalty factor, ξ_i, ξ_i^*, are the empirical risk. ε is the width of the band area, $i = 1, 2, 3, \cdots, n$. is the number of training data.

As shown in Equation (4), Cherkassky and Ma [25] proposed the Radial basis function (RBF) kernel, defined as follows:

$$\phi(x) = exp\left(-\gamma \left\|x_i - y_i\right\|^2\right), \quad \gamma = \frac{1}{2\delta^2} \tag{4}$$

where the γ parameter defines how far the influence of a single training example reaches. Since the RBF is suitable for solving forecast problems, we select the RBF as the kernel function in this work.

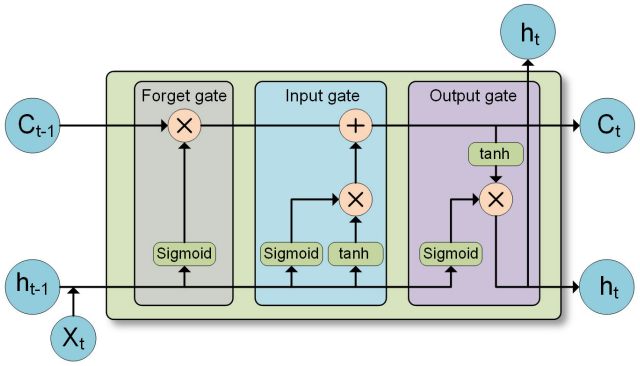

FIGURE 1. Basic structure of a long-short term memory network.

B. LONG SHORT-TERM MEMORY

Long short-term memory neural networks are a special type of recurrent neural networks (RNNs) [26], which is specially designed to solve the long-term dependence problem of general RNN. Due to its structure (see Fig. 1), LSTM is suitable for analyzing events with long delays in time series.

A LSTM unit consists of a memory cell and three gates. The core of LSTMs is the cell state, which is represented by a horizontal line running through the cell. The cell state is like a conveyor belt. It runs through the entire cell; however, it has only a few minor linear interactions, which ensure that the information flowing through the entire RNNs does not change. The memory cell state is carefully regulated by three gates, i.e., the forget gate, input gate, and output gate. It is composed of a sigmoid neural net layer and a pointwise multiplication operation.

The sigmoid layer outputs a value between 0 and 1, describing how much of each component should be let through. The value "0" means "nothing can go through". Additionally, the value "1" means "everything can go through".

The LSTM unit has a forget gate that decides what information is to be thrown away from the cell state. As shown in Fig. 1, the forget gate is composed of a hidden layer h_{t-1}, an input of the current layer X_t, and a cell state C_{t-1}. The latter outputs a value between 0 and 1 for each number in the cell state C_{t-1}. The value "0" indicates "completely forget it" while the value "1" indicates "completely retain it". The forget gate is given by:

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

An important part of the LSTM unit is the input gate that decides what information can be kept in the cell state. When the input gate is closed, no more information can get into the memory cell. Because of this unique function, the memory cell can maintain data until they need to be updated. It has two sections. The one, a sigmoid layer called the "input gate layer" i_t decides which values can be updated. The other, a hyperbolic tangent (tanh) layer creates a new candidate vector value $\tilde{C}_t$ that could be added to the state. The input gate updates the old cell state C_{t-1} into the new cell state C_t by forgetting the things that we decided to forget earlier, then plus the things we decided to remain $i_t * \tilde{C}_t$. The input gate's Equations (6-8) are as follows:

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right) \tag{6}$$
$$\tilde{C}_t = \tanh\left(W_C \cdot [h_{t-1}, x_t] + b_C\right) \tag{7}$$
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{8}$$

Finally, the output gate of the LSTM unit decides which information will be output. The sigmoid layer decides what parts of the cell state need to be output. Then, the output O_t comes from a product of the cell state and tanh. Therefore, nothing else but the desired information could be transmitted by the output gate. Their expressions are as follows:

$$O_t = \sigma\left(W_O \cdot [h_{t-1}, x_t] + b_O\right) \tag{9}$$
$$h_t = O_t * \tanh\left(C_t\right) \tag{10}$$

C. PARTICLE SWARM OPTIMIZATION

Particle Swarm Optimization (PSO) is a population-based metaheuristic optimization technology, proposed by Eberhart and Kennedy in 1995 [27]. The PSO algorithm imitates the swarm behavior of insects, animals, birds, fishes, and so on. They cooperatively search for food. Each member of the swarm constantly changes its search style by learning from its own experience and the experience of other members. PSO makes few or no assumptions about the problem being optimized and can search very large spaces of candidate solutions.

The PSO algorithm regards each individual as a "particle", all particles in the swarm have a fitness value determined by the fitness function $f(x)$, and each particle also has a velocity that determines the direction and distance of their flight. Furthermore, particles follow the current optimal particle and search in the solution space. The PSO is initialized as a group of random particles, and then the optimal position is found through iteration. In each iteration, the particle updates itself by tracking two best values; the first one is the particle's own best position, for example, $pbest_{ij}$. All particles in the swarm are able to share the information of the best point achieved so that the second one is the global best position called $gbest_{ij}$. Suppose a swarm with particles M in dimension N, the i_{th} particle is associated with two vectors, $X_{ij} = \left[x_{i1}, x_{i2}, x_{i3}, \cdots, x_{ij}\right]$ is a position vector and $V_{ij} = \left[v_{i1}, v_{i2}, v_{i3}, \cdots, v_{ij}\right]$ is a velocity vector. The velocity vector is updated as:

$$V_{ij}^{t+1} = \omega V_{ij}^{t} + c_1 r_1 \left(pbest_{ij} - X_{ij}^{t}\right) + c_2 r_2 \left(gbest_{ij} - X_{ij}^{t}\right) \tag{11}$$

where parameter $i = 1, 2, 3, \cdots, M$ represents the i_{th} particle, parameter $j = 1, 2, 3, \cdots, N$ represents the dimension of the problem space, and parameter t represents the number of iterations. In Equation (11), parameter ω represents the inertia weight factor that is a constant between 0 and 1; c_1, c_2 represent the individual cognition and social learning parameters, respectively. c_1, c_2 are set to 2 in

standard PSO. r_1 and r_2 are random values in the range of [0, 1]. The position of a particle will be updated by adding a velocity vector V_{ij}^{t+1} to the current position vector X_{ij}^{t}, i.e.

$$X_{ij}^{t+1} = X_{ij}^{t} + V_{ij}^{t+1} \tag{12}$$

where X_{ij}^{t+1}, V_{ij}^{t+1} represents the new position vector and the velocity vector at the next iteration, respectively.

D. PSO-LSTM MODEL

As a metaheuristic optimization algorithm, PSO can be used to optimize the performance of LSTM. Specifically, PSO for parameter selection can reduce computation time and increase forecast accuracy. In this paper, particles are defined as a four-dimensional variable that represent the number of iterations, the two hidden layer nodes of LSTM model, and the learning rate, respectively. The first three parameters are random integers in the range of [1, 200]. The last one is a random value in the range of [0.001,0.01]. The parameters $c_1, c_2, r_1, r_2, \omega$ are initialized as values 1.5, 1.5, 0.8, 0.3, 0.6, respectively. PSO-LSTM algorithm ends once the maximum number of iterations is reached.

III. PROPOSED ALGORITHM

A. IPSO-LSTM MODEL

As the parameters of LSTM are usually set heuristically, they are unlikely to be optimal. Recent advances indicate that the parameters of LSTM can be updated dynamically using PSO algorithm [28], [29]. However, PSO algorithm often falls into local optimum during iteration. To improve the diversity of the swarm in the PSO, we propose a novel IPSO algorithm to avoid premature convergence. Specifically, we improve the inertia weight of the standard PSO algorithm and propose an adaptive mutation factor. The IPSO is used to optimize the parameters of the LSTM and decrease the subjective influence of manually chosen parameters.

B. IMPROVED INERTIA WEIGHT

According to previous work, the inertia weight value is one of the most important adjustable parameters of the PSO model. A smaller inertia weight enhances exploitation while a larger inertia weight enhances exploration. Although the classical time-varying inertia weight is easy to implement, it cannot offer outstanding performance since different problems have their own distinctive characteristics. In this paper, the non-linear hyperbolic tangent function is used to balance the exploitation and exploration. The novel inertia weight is given by:

$$\omega_t = \omega_{\max} - (\omega_{\max} - \omega_{\min}) * \tanh\left(\frac{\pi}{4} * \frac{t}{\max\_iter}\right) \tag{13}$$

where ω_t is the improved inertia weight, $\omega_{\min}$ is the minimum inertia weight, and $\omega_{\max}$ is the maximum inertia weight, respectively. t is the current iteration and max _iter is the maximum number of iterations. ω_t is used to replace ω in Equation (11). Each particle inertia weight ω_t is updated independently according to Equation (13).

Since the inertial weight and the iteration time are negatively correlated, the particles have a relatively large inertial weight to promote the swarm diversity at the beginning of the evolutionary process. As the number of iterations increases, the inertia weight decreases non-linearly. The decreasing inertia weight can significantly enhance the swarm's convergence. Additionally, all the swarm particles have varying inertia weights since each inertia weight is updated by itself. Therefore, some particles can search locally, while the other particles are focused on global search. The non-linearly decreasing inertial weight in our approach is able to improve the performance of the PSO model by balancing the local and global search abilities.

C. ADAPTIVE MUTATION FACTOR

Although the non-linearly decreasing inertial weight can balance the local and global search capabilities of PSO, the particles can still get caught in local optima. Since the value of *gbest* is probably still a local optimum before the particles lose diversity, it is difficult to achieve global convergence only by using the above update strategy. Previous research has demonstrated that a suitable mutation operator was conducive to find global optimum. Zhang *et al.* [30] proposed a novel method that employs adaptive disturbance to balance the diversity and convergence speed of the swarm and to improve the global convergence. In this subsection, to address the issue of premature convergence and to increase the diversity of the particles, a new adaptive mutation factor is designed to enhance the optimization capability of PSO according to the distance between each particle and the optimal particle.

The adaptive mutation factor α_{mf} is closely related to the iteration number, and is defined as

$$\alpha_{mf} = 0.3 * \frac{t}{\max\_iter} + 0.7 \tag{14}$$

where α_{mf} represents the adaptive mutation factor that is set to be between (0.7, 1]. t is the current iteration number and max _*iter* is the maximum number of iterations. We see that α_{mf} increases as the iteration increases.

Mutation allows the position of a particle to be perturbed to escape from a local optimum. As the number of iterations increases, the probability of mutation should decrease. The new method devises a random value α for mutation operation. Let α be a random number generated from a [0,1] uniform distribution for a particle. The mutation operation is performed as follows:

1) When $\alpha > \alpha_{mf}$

The particles mutate. This increases their chance to escape from local optima. The IPSO-LSTM model randomly generates new hyperparameters of the LSTM network: the number of iterations, learning rate and two hidden layer nodes of the LSTM network. The model is updated by iteration optimization until the optimal values are found.

2) When $\alpha \leq \alpha_{mf}$

No mutation is performed on the particles and the IPSO model continues to update the *pbest* and the *gbest*

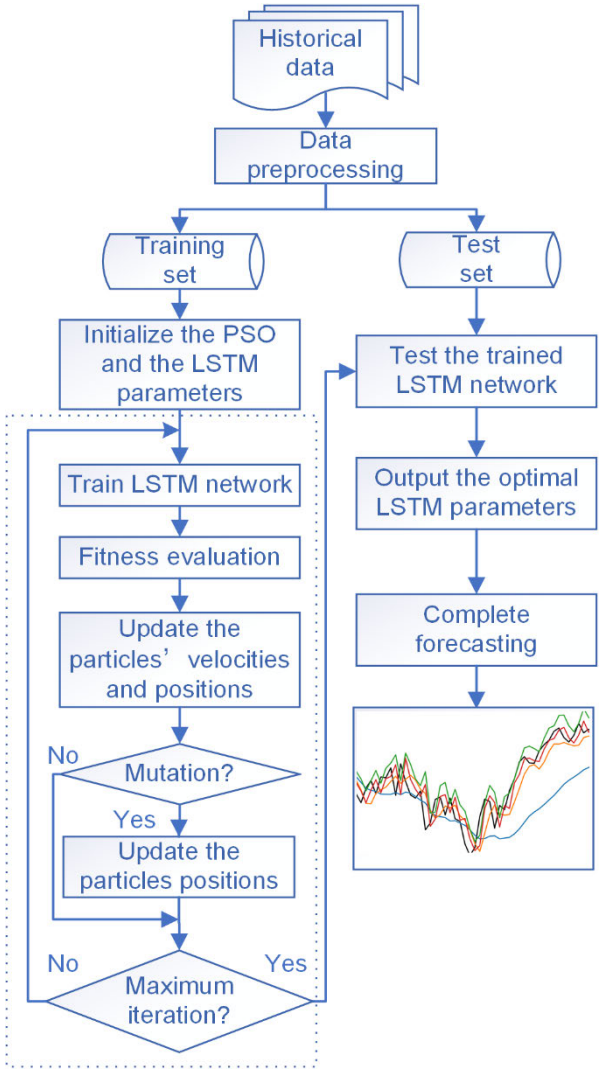

FIGURE 2. Flowchart of the IPSO-LSTM model for stock index forecasting.

through traditional PSO. The model is updated by iteration optimization until the optimal values are found.

D. THE PROPOSED FRAMEWORK OF IPSO-LSTM

In this work, the LSTM neural network is optimized by the improved PSO model, which is used to forecast the stock indices' movement. The overall process is shown in Fig. 2. In the IPSO-LSTM model, the first stage is data preprocessing that includes data splitting and data normalization. In the second stage, the IPSO algorithm is used to search for the optimal parameters of the LSTM network. Finally, the LSTM network with the optimal parameters is used to forecast the results of the stock index. The steps are:

Step 1: Normalize the stock index from the daily closing price into a real number in the range of [0, 1] using Min-Max normalization method. The historical data are then split into two groups: the training dataset and the test dataset.

Step 2: Initialize the IPSO parameters that include the population size, particle dimension, number of iterations, learning factor, particle position, particle velocity, inertia weight. Initialize the LSTM parameters that include the number of nodes in each hidden layer of the network, epoch, batch size and values of the look-back period.

Step 3: Use the training dataset to train the LSTM model. The right hyperparameters in the LSTM model are optimized by the IPSO algorithm with improved inertia weight and adaptive mutation factor.

Step 3.1: Train the LSTM network. The mean square error (MSE) is selected as the loss function. Update the loss function by training iterations.

Step 3.2: Evaluate fitness of each particle, determine the individual optimal fitness value and the global optimal fitness value.

Step 3.3: According to the Eq. (11) - (13), the velocity and position of particles are updated, respectively. In the iterative process, the velocity and position of the particle are continuously updated according to the two optimal values. When the adaptive mutation is applied, the particles will mutate to increase the chance of searching new areas.

Step 3.4: The optimal LSTM hyperparameters are recorded when the end conditions of the IPSO are met. Otherwise, go to Step 3.1 to continue the iteration.

Step 4: The trained model is tested by using the test dataset.

Step 5: Through above processing, the proposed method can find the best hyperparameters for the LSTM model.

Step 6: The optimal LSTM model is used for stock indices forecast. The forecasting results are evaluated by four performance metrics.

The models are implemented on a PC (Intel Core i7-10510U CPU at 1.8 GHz, 16Gbyte RAM, and GeForce MX250 GPU). The development environment is Python 3.7, PyCharm Edu 2020.1 IDE on a Windows 10 operating system. These models are implemented under the TensorFlow machine learning platform.

IV. EXPERIMENTS

A. DATASET

In this study, we select the Australian stock market (ASM) as our main focus, where the S&P/ASX200 index (ASX200 or XJO) is considered the benchmark for ASM performance, which is compiled based on the 200 largest ASX listed stocks. The ASM is a mature, healthy, and high return market. Since 1900, the ASM has returned an average over 13% per annum [31]. The average return in investment is higher than other well-known stock markets, such as New York Stock Exchange, Nasdaq, Tokyo Stock Exchange. The experimental dataset is comprised of the daily historical data in the past 10 years from 1 September 2010 to 31 August 2020. All data are derived from Yahoo! Finance, as shown in Fig. 3.

B. DATA PREPROCESSING

Data preprocessing is a crucial step in data analytics, high quality data leads to better models and predictions. The S&P/ASX200 index closing prices are normalized to real

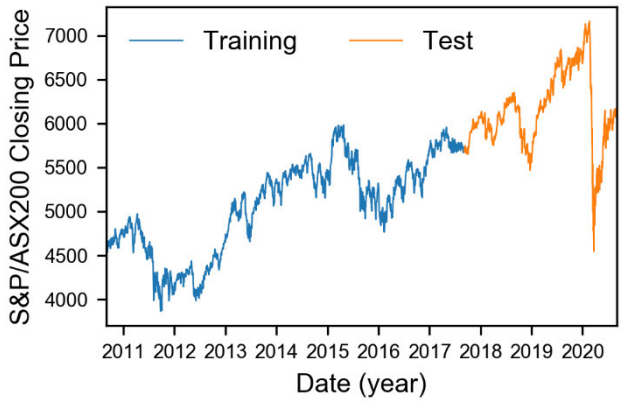

FIGURE 3. The S&P/ASX200 index from September 2010 to August 2020.

numbers in the range of [0, 1] by:

$$x'(t) = \frac{x(t) - x_{\min}}{x_{\max} - x_{\min}} \tag{15}$$

where t represents each trading day, $x(t)$ represents raw input data that is the closing price of the daily data, $x'(t)$ represents the normalized input data that is normalized from $x(t)$, $x_{\min}$ and $x_{\max}$ represent the smallest and largest value of the raw data, respectively.

After data preprocessing, the raw dataset is split into two separate sets in various ratios. As shown in Fig. 3, the blue line represents the first 70% of daily historical data and they form the training set. The orange line represents the remaining 30% of the data that are used as the test set.

C. PERFORMANCE METRICS

Measuring forecast accuracy is not an easy task as there is no one-size-fits-all metric. To appropriately evaluate the forecasting capability of each method, the following four common metrics are used to measure accuracy: root mean square error (RMSE), mean absolute error (MAE), mean absolute percentage error (MAPE), and R2_score (R^2). The four performance metrics are expressed by the following Equations (16-19):

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{t=1}^{N}\left(y_{test}(t) - \hat{y}_{test}(t)\right)^2} \tag{16}$$

$$\text{MAE} = \frac{1}{N}\sum_{t=1}^{N}\left|y_{test}(t) - \hat{y}_{test}(t)\right| \tag{17}$$

$$\text{MAPE} = \frac{100\%}{N}\sum_{t=1}^{N}\left|\frac{y_{test}(t) - \hat{y}_{test}(t)}{y_{test}(t)}\right| \tag{18}$$

$$R^2 = 1 - \frac{MSE\left(y_{test}(t) - \hat{y}_{test}(t)\right)}{\text{Var}(y_{test}(t))} \tag{19}$$

where N indicates the number of samples. $y_{test}(t)$ and $\hat{y}_{test}(t)$ represent the true value and the forecast value, respectively. Var indicates the variance, MSE is the mean squared error.

To determine the LSTM model parameters, six widely accepted settings are used. The epochs and the number of nodes in each of the two hidden layers of the LSTM model are random integers sampled from the range [1, 200]. The learning rate η is randomly generated within the range of [0.001, 0.01]. Basic parameter settings are summarized in Table 1.

TABLE 1. Basic parameter settings of the four different models.

Model	Parameter Settings
SVR	Kernel = RBF, C = 1000, gamma = 0.001
LSTM	Node1 = 100, Node2 = 20, Epoch = 64, η = 0.001, Batch_size = 64
PSO-LSTM	Node1, Node2, Epoch∈ [1-200], η∈[0.001,0.01], Batch_size = 64, ω = 0.6
IPSO-LSTM	Node1, Node2, Epoch∈ [1-200], η∈[0.001,0.01], Batch_size = 64, ω_t ∈[0.6, 0.8]

TABLE 2. Comparison of the forecasting results corresponding to the different evaluation metrics.

Look-back (days)	Model	MAPE(%)	RMSE	MAE	R^2
5	SVR	4.129656	271.015312	256.496323	0.610411
5	LSTM	1.844114	151.139208	114.509607	0.879149
5	PSO-LSTM	**0.736593**	**68.982782**	**43.806668**	**0.974759**
5	IPSO-LSTM	1.154356	87.289314	70.668976	0.959585
10	SVR	2.323514	166.202245	141.565111	0.853443
10	LSTM	2.361535	197.375997	146.561536	0.793866
10	PSO-LSTM	0.919290	75.770024	55.491421	0.969540
10	IPSO-LSTM	**0.753436**	**71.142792**	**44.683590**	**0.973147**
15	SVR	1.763938	151.808381	105.692894	0.877698
15	LSTM	2.534172	205.943853	156.631631	0.775552
15	PSO-LSTM	1.169543	88.019382	71.450168	0.958885
15	IPSO-LSTM	**0.897608**	**75.407053**	**54.505893**	**0.969824**
20	SVR	1.892891	159.915564	113.705858	0.864314
20	LSTM	1.621149	142.105609	99.297178	0.893108
20	PSO-LSTM	1.032172	84.722259	62.252169	0.961915
20	IPSO-LSTM	**0.723480**	**67.322435**	**43.133272**	**0.975952**
30	SVR	2.094065	172.659452	126.047232	0.841859
30	LSTM	2.330110	189.853135	143.722121	0.809311
30	PSO-LSTM	**0.759001**	69.949026	**45.087240**	0.974045
30	IPSO-LSTM	0.793433	**68.164295**	47.827836	**0.975352**
60	SVR	2.371200	193.432680	141.999597	0.801141
60	LSTM	2.243322	176.462373	137.524003	0.834987
60	PSO-LSTM	1.467592	106.111847	90.164892	0.940157
60	IPSO-LSTM	**0.865363**	**72.527546**	**52.346409**	**0.972043**

D. S&P/ASX200 FORECASTING

The entire dataset includes 2523 records in the form of S&P/ASX200 daily closing prices. The first 1767 data from 1 September 2010 to 5 September 2017 are used as the training dataset. The last 756 data from 6 September 2017 to 31 August 2020 are used as the test dataset. To evaluate the performance of each model, the forecasting results are evaluated by the four performance metrics.

The value of the look-back period indicates the number of previous days' data to be used in the forecasting. The data are the daily closing prices. The value of the look-back period is critical for achieving good forecast accuracy in a time series model.

To evaluate the performance of the four models under various look-back periods, we selected six different look-back periods, i.e., 5, 10, 15, 20, 30, and 60. The experimental results corresponding to different look-back periods are shown in Table 2 and Fig. 4. Each model is evaluated by six different look-back periods and four different performance metrics. To highlight the best forecasting results, the optimal values of each look-back period are identified in boldface.

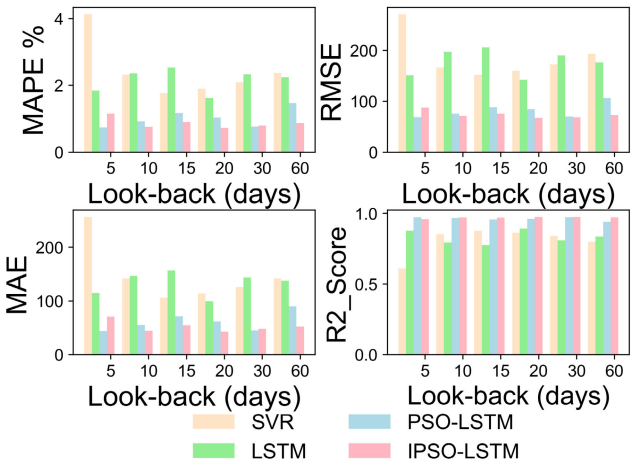

FIGURE 4. Evaluation results with the four metrics on the ASX200 index (look-back = 20 days).

From Table 2 and Fig. 4, it can be observed that the forecasting accuracy does not vary monotonically with look-back periods. The proposed model performs the best when look-back = 20 days. Hence, for subsequent experiments a look-back of 20 days is used.

It can be seen that MAPE, RMSE, MAE, and R^2 of the IPSO-LSTM with 20 days look-back are better than SVR by 61.78%, 57.90%, 62.07%, and 12.92%, respectively, whereas they are better than PSO-LSTM by 29.90%, 20.54%, 30.71%, and 1.46%, respectively, and better than LSTM by 55.37%, 52.63%, 56.56%, and 9.28%, respectively. The R^2 values for IPSO-LSTM are in the range of [0.9596, 0.9760], for PSO-LSTM they are in the range of [0.9402, 0.9748], and for LSTM they are in the range of [0.7756, 0.8791]. Table 2 also shows that although the LSTM forecasting can be significantly improved by PSO (with an improvement of MAPE, RMSE, MAE, and R^2 by 36.33%, 40.38%, 37.31%, and 7.70%, respectively), the IPSO-LSTM model can further improve the performance. These results show that the parameters of LSTM can be optimized by PSO, and furthermore by IPSO due to its ability to avoid getting trapped in local optima, and the improved LSTM has better forecasting performance on the short-term and medium-term stock indices.

To verify the performance of the IPSO-LSTM model over that of PSO-LSTM, the following experiments are performed: the LSTM model with two hidden layers is optimized by (1) IPSO, and (2) standard PSO (PSO). The changes in the number of the hidden layer nodes and the learning rate vs the number of training iteration of the LSTM network are shown in Fig.5. The orange lines represent the IPSO-LSTM model hyperparameters, whereas the green lines represent the PSO-LSTM model hyperparameters. The X-axis represents the number of iterations for IPSO and PSO. The Y-axis on the left subgraph of Fig. 5 refers to the number of LSTM iterations, the number of nodes in the first hidden layer (Node1), and the number of nodes in the second hidden layer (Node2). The Y-axis on the right subgraph of Fig. 5 refers to the learning rate of the LSTM networks. Although PSO-LSTM shows faster convergence at the initial stage, more accurate results cannot be produced in the later stage. The use of an adaptive mutation factor allows IPSO-LSTM to escape from local optimum, thereby allowing it to outperform PSO-LSTM and other models.

Fig. 6 shows the forecasting of the ASX200 index by the four models for a period of 90days (from 23 October 2018 to 3 March 2019). Clearly, the forecasting given by IPSO-LSTM is the closest to the true value.

The proposed IPSO-LSTM model is compared with the four models for the ASX200 index from the period 6 September 2017 to 31 August 2020 and the results are shown in Fig.7. The SVR performs the worst among all models. The IPSO-LSTM significantly outperforms the other models.

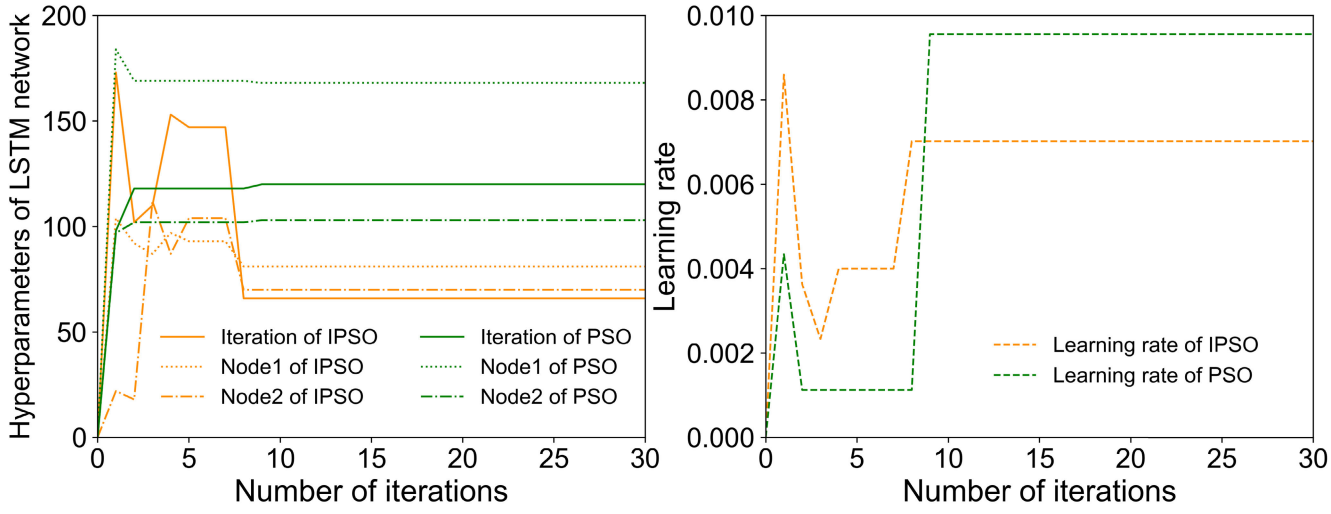

FIGURE 5. Hyperparameters optimization with IPSO-LSTM and PSO-LSTM on the ASX200 index (look-back = 20 days).

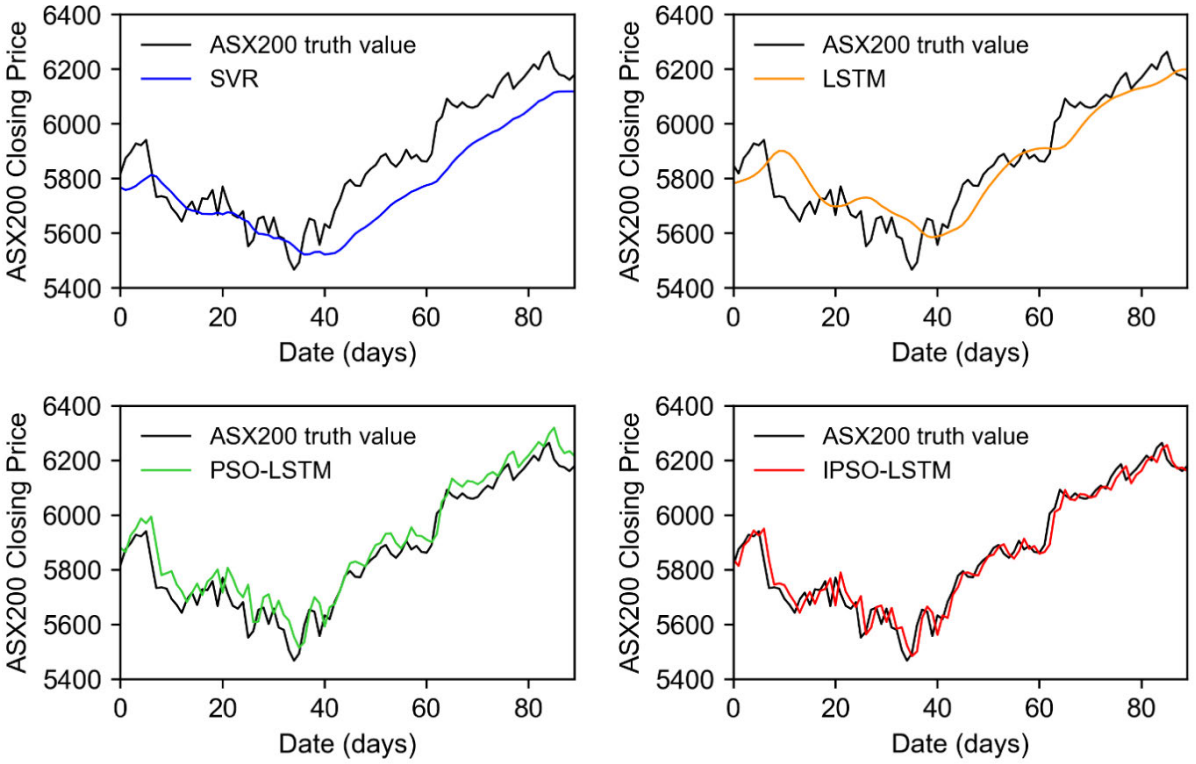

FIGURE 6. Forecasting results of different models on the S&P/ASX200 index for the 90 days test period (look-back = 20 days).

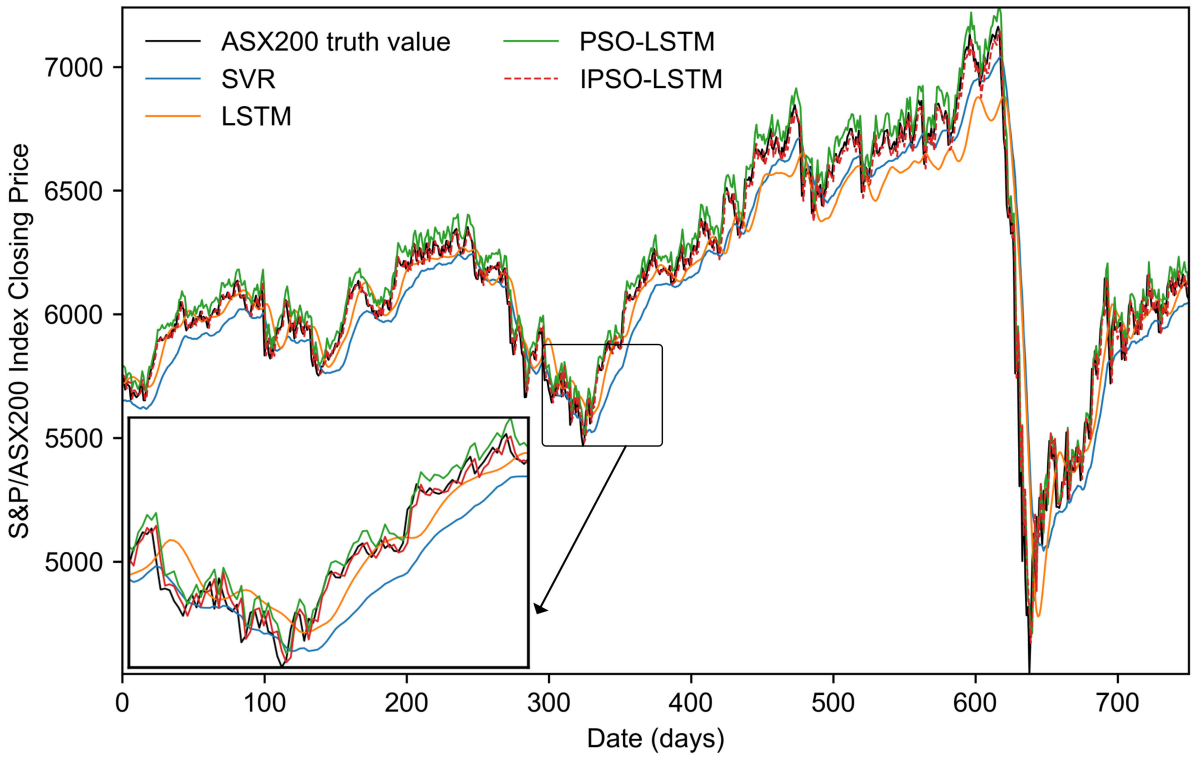

FIGURE 7. Forecasting results of different models on the S&P/ASX200 index for the whole test set (look-back = 20 days).

It can be seen from the partially enlarged subgraph in Fig.7 that the IPSO-LSTM forecasting is very close to the true curve.

To further verify the forecasting capability of the IPSO-LSTM model, we compare it with the PSO-LSTM and the other improved variable inertia weight methods of

TABLE 3. The basic parameter settings and experimental results corresponding to the different models.

Model	Parameter Settings					Evaluation Metrics			
	Inertia weight	Learning factors	Epochs	Nodes	Look-back (days)	MAPE (%)	RMSE	MAE	R^2
Wang *et al.* [28]	ω_{max} = 0.9, ω_{min} = 0.4	C_1, C_2 = 2	[50,350]	Node1∈[1,50], Node2∈[1,50]	20	1.46348	108.295384	90.617452	0.937774
Shao *et al.* [29]	ω_{max} = 0.8, ω_{min} = 0.2	C_1, C_2 = 2	50	Node1 = 4, Node2 = 2	20	1.468021	117.719496	91.084410	0.926458
PSO-LSTM	ω = 0.6	C_1, C_2 = 1.5	[1,200]	Node1 = 100, Node2 = 20	20	1.032172	84.722259	62.252169	0.961915
IPSO-LSTM	ω_{max} = 0.8, ω_{min} = 0.6	C_1, C_2 = 1.5	[1,200]	Node1∈[1,200], Node2∈[1,200]	20	**0.72348**	**67.322435**	**43.133272**	**0.975952**

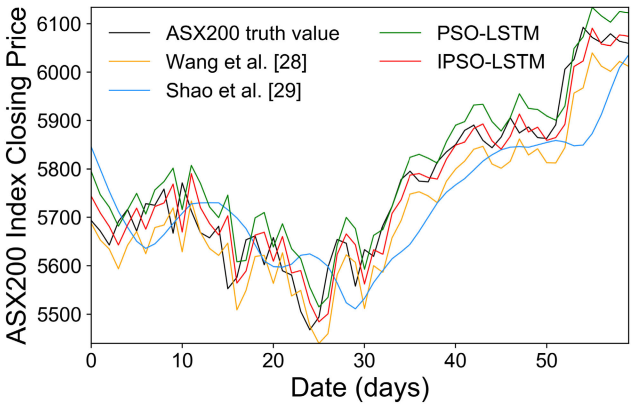

FIGURE 8. Forecasting results of different deep learning models on the S&P/ASX200 index.

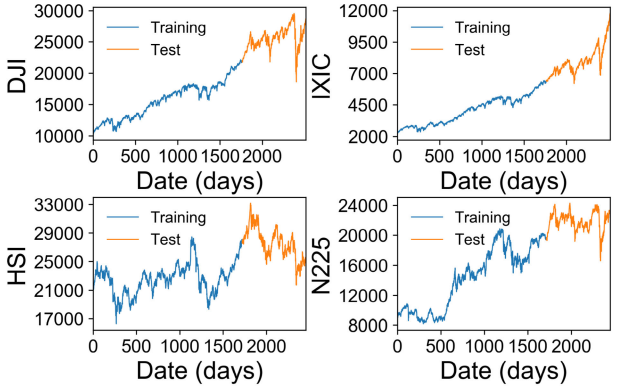

FIGURE 9. The four major stock market indices daily closing price from September 2010 to August 2020.

Wang *et al.* [28] and Shao *et al.* [29]. Except for the look-back period, the other parameter settings for those models are inherited from the original papers. The forecasting results are shown in Fig. 8. The comparison results are shown in Table 3.

According to the performance metrics MAPE, RMSE, MAE, and R^2, the proposed model still achieves the best performance among the four models. Table 3 shows that Wang *et al.* [28] and Shao *et al.* [29] could improve the forecasting results of LSTM. However, the proposed IPSO-LSTM model still has the best performance over all four models.

E. COMPARISON USING DIFFERENT STOCK INDICES

To further verify the robustness and reliability of the proposed model, we run experiments on four well-known indices: DJI, IXIC, HSI, and N225. The Dow Jones Industrial Average Index (DJI) is the stock market index that measures the stock performance of 30 large companies listed on the stock exchange in the United States. Nasdaq Composite Index (IXIC) is the index for the Nasdaq stock market. Hang Seng Index (HSI) is the main stock market performance index in Hong Kong. Nikkei 225 Index (N225) measures the performance of 225 large companies listed on Tokyo Stock Exchange in Japan. The dataset time frame is from 01 September 2010 to 31 August 2020 which is the same as the selected ASX200 time frame. To conduct the experiments, the first 70% of the samples are used to train the model, and the remaining 30% are used to test the model. Table 4 lists all five stock indices used in our experiment.

The daily closing price charts of the four stock indices are shown in Fig. 9. The blue line on the left indicates the training set. The orange line on the right indicates the test set. On each subgraph, the x-axis indicates the date from 1 September 2010 to 31 August 2020. The y-axis indicates the daily closing price of each index.

The parameter setting of the IPSO-LSTM model is the same as those used in the previous ASX200 experiment. The results of the four stock indices by IPSO-LSTM are shown in Table 5 and Fig. 10.

Table 5 shows the forecasting results of various stock indices based on the four evaluation criteria. As shown in Fig. 10, the blue line represents the observed true value. The orange line represents the forecasting value by IPSO-LSTM.

TABLE 4. The details of the five selected stock indices.

Index	Dataset	#Data	Training set	#Data	Test set	#Data
DJI	Sep 2010~Aug 2020	2517	Sep 01, 2010~Aug 30, 2017	1762	Aug 31, 2017~Aug 31, 2020	755
IXIC	Sep 2010~Aug 2020	2517	Sep 01, 2010~Aug 30, 2017	1762	Aug 31, 2017~Aug 31, 2021	755
HSI	Sep 2010~Aug 2020	2458	Sep 01, 2010~Sep 04, 2017	1721	Sep 05, 2017~Aug 31, 2018	737
N225	Sep 2010~Aug 2020	2446	Sep 01, 2010~Aug 28, 2017	1713	Aug 29, 2017~Aug 31, 2019	733
XJO	Sep 2010~Aug 2020	2523	Sep 01, 2010~Sep 05, 2017	1767	Sep 06, 2017~Aug 31, 2020	756

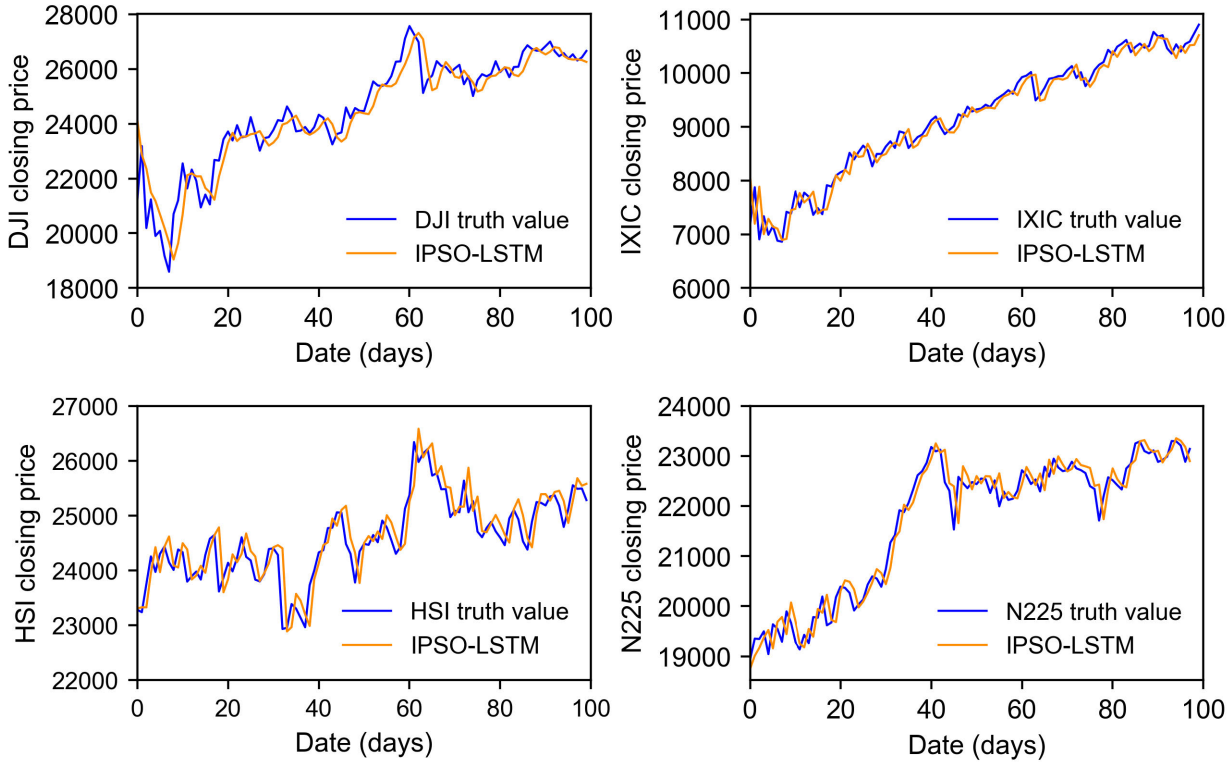

FIGURE 10. Forecast capability of the IPSO-LSTM model on four different indices.

TABLE 5. Comparison of the experimental results corresponding to different stock indices and evaluation metrics.

Index	Look-back (days)	Model	MAPE (%)	RMSE	MAE	R^2
DJI	20	IPSO-LSTM	1.062803	390.217	266.219	0.94789
IXIC	20	IPSO-LSTM	0.997694	123.219	79.6235	0.98673
HSI	20	IPSO-LSTM	1.019524	367.745	277.472	0.97118
N225	20	IPSO-LSTM	0.908136	276.048	194.429	0.95186
XJO	20	IPSO-LSTM	0.72348	67.3224	43.1333	0.97595

The experimental results show that the forecasting curve is very close to the true value for all four major indices, indicating the generality and robustness of the proposed IPSO-LSTM algorithm to different stock markets.

Although we have seen that the LSTM model performs well in forecasting stock indices, there are also some limitations. LSTM model requires a lot of resources and time to train, and these can be a barrier for real-world applications. On the other hand, it is well-known that there are other external factors that affect stock volatility. Hence, sophisticated models that could take external factors into consideration would be needed to further improve the forecasting performance.

V. CONCLUSION

This study was concerned with a novel IPSO-LSTM hybrid model to forecast stock market indices. We showed that by using a novel inertia weight and an adaptive mutation factor in the PSO optimization, we are able to search for better hyperparameters for the LSTM network, which contribute to improving forecast accuracy. We evaluated the performance of the proposed IPSO-LSTM algorithm on five major stock market indices (ASX200, DJI, IXIC, HSI, N225) using four performance metrics (RMSE, MAE,

MAPE, and R^2), and compared with well-known algorithms or SOTA deep-learning models, such as SVR and PSO optimized LSTMs. The comparative study showed that the proposed IPSO-LSTM model has the best performance and it generalized well to different stock market indices. Future study will involve using other heuristics, meta-heuristics, or hybrid algorithms for feature selection and parameter optimization. We will also explore the correlations of different stock markets in index forecasting. Moreover, forex, commodities, bonds, and futures are popular financial products that attract hot money and their volatility may have an effect on each other. Consequently, it would be interesting to investigate whether and how they are correlated with stock market movement or trend. We plan to develop a novel model which has adaptive attention weights that can be updated according to the impact of different financial products on market volatility using some sort of attention mechanism [32].

ACKNOWLEDGMENT

The authors would like to thank the Editor-in-Chief and the anonymous peer-reviewers for their valuable comments and suggestions. In addition, they also thank Jet Brains for IDE PyCharm Edu and Yahoo! Finance for raw data.

CONFLICTS OF INTEREST

The authors declare that there is no conflict of interest.

SUPPORTING INFORMATION

The experimental raw data has been uploaded to the IEEE DataPort (http://dx.doi.org/10.21227/vvr8-8g25).

REFERENCES

[1] F. E. H. Tay and L. Cao, "Application of support vector machines in financial time series forecasting," *Omega*, vol. 29, no. 4, pp. 309–317, Aug. 2001.
[2] K.-J. Kim, "Financial time series forecasting using support vector machines," *Neurocomputing*, vol. 55, nos. 1–2, pp. 307–319, Sep. 2003.
[3] W. Huang, Y. Nakamori, and S.-Y. Wang, "Forecasting stock market movement direction with support vector machine," *Comput. Oper. Res.*, vol. 32, no. 10, pp. 2513–2522, Oct. 2005.
[4] A. Hossain and M. Nasser, "Comparison of the finite mixture of ARMA-GARCH, back propagation neural networks and support-vector machines in forecasting financial returns," *J. Appl. Statist.*, vol. 38, no. 3, pp. 533–551, Mar. 2011.
[5] A. Sheta, S. E. M. Ahmed, and H. Faris, "A comparison between regression, artificial neural networks and support vector machines for predicting stock market index," *Int. J. Adv. Res. Artif. Intell.*, vol. 4, no. 7, pp. 55–63, Jul. 2015.
[6] H. Qu and Y. Zhang, "A new kernel of support vector regression for forecasting high-frequency stock returns," *Math. Problems Eng.*, vol. 2016, pp. 1–9, Jan. 2016.
[7] C.-Y. Yeh, C.-W. Huang, and S.-J. Lee, "A multiple-kernel support vector regression approach for stock market price forecasting," *Expert Syst. Appl.*, vol. 38, no. 3, pp. 2177–2186, Mar. 2011.
[8] J.-C. Hung, "Fuzzy support vector regression model for forecasting stock market volatility," *J. Intell. Fuzzy Syst.*, vol. 31, no. 3, pp. 1987–2000, Aug. 2016.
[9] W. Jujie and Q. Danfeng, "An experimental investigation of two hybrid frameworks for stock index prediction using neural network and support vector regression," *Econ. Comput. Econ. Cybern. Stud. Res.*, vol. 52, no. 4, pp. 193–210, Dec. 2018.
[10] K. Chen, Y. Zhou, and F. Dai, "A LSTM-based method for stock returns prediction: A case study of China stock market," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Oct. 2015, pp. 2823–2824.
[11] D. M. Q. Nelson, A. C. M. Pereira, and R. A. D. Oliveira, "Stock market's price movement prediction with LSTM neural networks," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Anchorage, AK, USA, May 2017, pp. 1419–1426.
[12] W. Bao, J. Yue, and Y. Rao, "A deep learning framework for financial time series using stacked autoencoders and long-short term memory," *PLoS ONE*, vol. 12, no. 7, Jul. 2017, Art. no. e0180944.
[13] T. Fischer and C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," *Eur. J. Oper. Res.*, vol. 270, no. 2, pp. 654–669, Oct. 2018.
[14] C.-J. Lu, "Hybridizing nonlinear independent component analysis and support vector regression with particle swarm optimization for stock index forecasting," *Neural Comput. Appl.*, vol. 23, nos. 7–8, pp. 2417–2427, Dec. 2013.
[15] W. Shen, X. Guo, C. Wu, and D. Wu, "Forecasting stock indices using radial basis function neural networks optimized by artificial fish swarm algorithm," *Knowl.-Based Syst.*, vol. 24, no. 3, pp. 378–385, Apr. 2011.
[16] X. Xia, L. Gui, G. He, B. Wei, Y. Zhang, F. Yu, H. Wu, and Z.-H. Zhan, "An expanded particle swarm optimization based on multi-exemplar and forgetting ability," *Inf. Sci.*, vol. 508, pp. 105–120, Jan. 2020.
[17] A. Nickabadi, M. M. Ebadzadeh, and R. Safabakhsh, "A novel particle swarm optimization algorithm with adaptive inertia weight," *Appl. Soft Comput.*, vol. 11, no. 4, pp. 3658–3670, Jun. 2011.
[18] Y. Ji, H. J. Fu, and K. H. Chen, "Numerical optimization for vibration and noise of the wheel based on PSO-GA method," *J. Vibroeng.*, vol. 19, no. 6, pp. 4609–4629, Sep. 2017.
[19] Q. Huang, J. Yang, X. Feng, A. W.-C. Liew, and X. Li, "Automated trading point forecasting based on bicluster mining and fuzzy inference," *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 2, pp. 259–272, Feb. 2020.
[20] Y. Guo, S. Han, C. Shen, Y. Li, X. Yin, and Y. Bai, "An adaptive SVR for high-frequency stock price forecasting," *IEEE Access*, vol. 6, pp. 11397–11404, 2018.
[21] X. Xia, L. Gui, F. Yu, H. Wu, B. Wei, Y.-L. Zhang, and Z.-H. Zhan, "Triple archives particle swarm optimization," *IEEE Trans. Cybern.*, vol. 50, no. 12, pp. 4862–4875, Dec. 2020.
[22] Y. Zhang, D.-W. Gong, and J. Cheng, "Multi-objective particle swarm optimization approach for cost-based feature selection in classification," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 14, no. 1, pp. 64–75, Jan. 2017.
[23] V. N. Vapnik, *The Nature of Statistical Learning Theory*, 1st ed. New York, NY, USA: Springer, 1995, pp. 151–155.
[24] H. Drucker, C. J. C. Burges, L. Kaufman, A. Smola, and V. Vapnik, "Support vector regression machines," in *Proc. 9th Int. Conf. Neural Inf. Process. Syst. (NIPS)*, Dec. 1996, pp. 155–161.
[25] V. Cherkassky and Y. Ma, "Practical selection of SVM parameters and noise estimation for SVM regression," *Neural Netw.*, vol. 17, no. 1, pp. 113–126, Jan. 2004.
[26] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[27] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, Perth, WA, Australia, Nov./Dec 1995, pp. 1942–1948.
[28] P. Wang, J. Zhao, Y. Gao, M. A. Sotelo, and Z. Li, "Lane work-schedule of toll station based on queuing theory and PSO-LSTM model," *IEEE Access*, vol. 8, pp. 84434–84443, 2020.
[29] B. Shao, M. Li, Y. Zhao, and G. Bian, "Nickel price forecast based on the LSTM neural network optimized by the improved PSO algorithm," *Math. Problems Eng.*, vol. 2019, pp. 1–15, Sep. 2019.
[30] Y. Zhang, D.-W. Gong, X.-Y. Sun, and N. Geng, "Adaptive bare-bones particle swarm optimization algorithm and its convergence analysis," *Soft Comput.*, vol. 18, no. 7, pp. 1337–1352, Jul. 2014.
[31] S. H. Williamson. (2020). *Annualized Growth Rate and Graphs of the DJIA, S&P500, and NASDAQ in the United States Between Any Two Dates*. MeasuringWorth. [Online]. Available: https://www.measuringworth.com/calculators/DJIA_SP_NASDAQ/
[32] T. Hollis, A. Viscardi, and S. Eun Yi, "A comparison of LSTMs and attention mechanisms for forecasting financial time series," 2018, *arXiv:1812.07699*. [Online]. Available: http://arxiv.org/abs/1812.07699

YI JI was born in Zhenjiang, Jiangsu, China, in 1981. He received the M.S. degree from Jiangsu University, China, in 2006, where he is currently pursuing the Ph.D. degree in computer. He is currently a full-time Associate Professor with the School of Electrical and Information Engineering, Jiangsu University. He has published more than ten papers in international journals and conference proceedings in the recent five years. His research interests include time series prediction, artificial intelligence, and signal processing.

ALAN WEE-CHUNG LIEW (Senior Member, IEEE) is currently an Associate Professor with the School of Information and Communication Technology, Griffith University, Australia. Prior to joining Griffith University in 2007, he was an Assistant Professor with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, and a Senior Research Fellow with the City University of Hong Kong. He is the author of two books and more than 200 book chapters, journal and conference papers, and holds two international patents. His research interests include machine learning and AI, computer vision, medical imaging, and bioinformatics. He has published extensively in these areas. He has engaged actively in professional activities, such as on the technical program committee of many conferences, on editorial boards, as an assessor for nationally competitive research grants, and a reviewer for many international conferences and journals. He is currently serving as an Associate Editor for IEEE TRANSACTIONS ON FUZZY SYSTEMS, and the *International Journal of Computational Intelligence Systems*.

LIXIA YANG (Member, IEEE) was born in Ezhou, Hubei, China, in 1975. He received the B.S. degree in physics from Hubei University, Wuhan, China, in 1997, and the Ph.D. degree in radio physics from Xidian University, Xi'an, China, in 2007. From 2010 to 2011, he was a Postdoctoral Research Fellow with the ElectroScience Laboratory (ESL), The Ohio State University. From 2015 to 2016, he was a Visiting Scholar with the Institute of Space Science, The University of Texas at Dallas, TX, USA. Since 2010, he has been an Associate Professor with the Communication Engineering Department, Jiangsu University, where he has been a Professor, a Ph.D. Supervisor, and the Chairman, since 2016. He is the author of a book, more than 100 articles, more than ten inventions, and holds four patents. His research interests include wireless communication technique, radio sciences, the computational electromagnetic, and the antenna theory and design in wireless communication systems. He is a member of the Editor Board of Radio Science Journal in China.