



# An adaptive particle swarm optimization-based hybrid long short-term memory model for stock price time series forecasting

Gourav Kumar<sup>1</sup> · Uday Pratap Singh<sup>2</sup> · Sanjeev Jain<sup>3</sup>

Accepted: 7 August 2022 / Published online: 26 August 2022  
© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2022

## Abstract

In this paper, we presented a long short-term memory (LSTM) network and adaptive particle swarm optimization (PSO)-based hybrid deep learning model for forecasting the stock price of three major stock indices such as Sensex, S&P 500, and Nifty 50 for short term and long term. Although the LSTM can handle uncertain, sequential, and nonlinear data, the biggest challenge in it is optimizing its weights and bias. The back-propagation through time algorithm has a drawback to overfit the data and being stuck in local minima. Thus, we proposed PSO-based hybrid deep learning model for evolving the initial weights of LSTM and fully connected layer (FCL). Furthermore, we introduced an adaptive approach for improving the inertia coefficient of PSO using the velocity of particles. The proposed method is an aggregation of adaptive PSO and Adam optimizer for training the LSTM. The adaptive PSO attempts to evolve the initial weights in different layers of the LSTM network and FCL. This research also compares the forecasting efficacy of the proposed method to the genetic algorithm (GA)-based hybrid LSTM model, the Elman neural network (ENN), and standard LSTM. Experimental findings demonstrate that the suggested model is successful in achieving the optimum initial weights and bias of the LSTM and FC layers, as well as superior forecasting accuracy.

**Keywords** Deep learning · Adaptive particle swarm optimization · Genetic algorithm · Long short-term memory network · Swarm intelligence · Technical analysis · Stock price time series

## 1 Introduction

Stock price time series is highly unpredictable as it exhibits irregular movements and possesses dynamic, highly volatile, and complex nonlinear behavior (Fama 1970). The economic growth of countries is affected by various financial activities and the stock market plays a critical role in the country's domestic as well as global economic conditions (Lin et al. 2012). The stock market is influenced

by several factors such as industrial growth, political instability, government policies, global economic position, psychology of investors, and many others factors (Menkhoff 1997). The stock price time series defined as:  $x_t = \{x_t \in \mathbb{R} | t = 1, 2, 3, \dots, N\}$  is the set of prices collected at regular time interval  $t$  (Pal and Kar 2022). The key focus of the stock price time series forecasting is to forecast the future values of the stock based on the past pattern present in the series itself. According to National securities depository limited (NSDL) (NSDL 2021) and Central depository services (India) limited (CDSL) (CDSL 2021) approximately 5.2% of the Indian population only invest in the stock market due to fear of loss, lack of knowledge, poor services from advisories and brokers and one most important factor that drives the market is uncertainty. Hence, there is a growing need to develop an intelligent system in order to model the uncertainty and nonlinearity in stock price time series. So, forecasting stock price is an important task for reducing the risk of the investor as well as the economic development of the country.

✉ Uday Pratap Singh  
usinghiitg@gmail.com

<sup>1</sup> School of Computer Science and Engineering, Shri Mata Vaishno Devi University, Katra, Jammu and Kashmir 182320, India

<sup>2</sup> School of Mathematics, Shri Mata Vaishno Devi University, Katra, Jammu and Kashmir 182320, India

<sup>3</sup> Department of Computer Science and Information Technology, Central University of Jammu, Samba, Jammu and Kashmir 181143, India

In the past few years, computational intelligence (CI) paradigms namely artificial neural networks (ANNs) (Shahvaroughi Farahani and Razavi Hajiagha 2021), genetic algorithm (GA), fuzzy logic (FL), and numerous nature-inspired optimization (Mahajan et al. 2021, 2022a) techniques namely particle swarm optimization (PSO), differential evolution (DE), artificial bee colony (ABC), bacterial foraging optimization (BFO), etc., have been combined to develop hybrid models for share price forecasting (Kumar et al. 2021a). CI is a growing computing approach that mimics the human brain's ability to generalize and memorize in the field of imprecision and uncertainty to solve difficult problems (Ibrahim 2016; Mahajan et al. 2022b). ANN can approximate the nonlinear relation present in data without making any assumptions, thus it is successfully used for stock price forecasting (Guresen et al. 2011). The most widely utilized artificial neural networks techniques for share price forecasting are recurrent neural networks (RNN) and feed-forward neural networks (FFNN) due to their ability to model nonlinearity and uncertainty (Atsalakis and Valavanis 2009; Liu and Wang 2012). Gong et al. (2018) applied the multiobjective evolutionary algorithm to obtain the optimal architecture of RNN for time series classification and estimation. Yan (2012) used a combination of generalized regression neural network models to automate the exploration for ANN parameters for time series forecasting. Gao et al. (2018) used various nature-inspired methods including PSO and GA to train the dendritic neuron model to solve multiple tasks. In recent studies, however, deep learning models have been widely used for stock price time series forecasting utilizing historical stock prices and technical indicators (Li and Bastos 2020). Convolution neural network (CNN) (Chen et al. 2020), recurrent neural network (RNN) (Berradi and Lazaar 2019), and deep belief network (DBN) (Zhang and Ci 2020) are three main paradigms of deep learning. Cao and Wang (2019) combined CNN with a support vector machine (SVM) to foresee the trend in the market and analyzed that the deep neural network can obtain good forecasting results in the financial time series prediction domain. Balaji et al. (2018) designed fourteen deep learning models based on CNN, long short-term memory (LSTM), gated recurrent unit (GRU), and extreme learning machine (ELM) to forecast 1-day ahead and 4-days ahead close price of all the stocks in S&P BSE-BANKEKX and concluded that deep learning models are effective in producing highly accurate forecasting results. Rather et al. (2015) created a hybrid method by combining linear models namely auto-regressive moving average and exponential smoothing and a nonlinear model viz. RNN to predict the stock returns. Further, in this study, the parameters of the proposed model are optimized by using GA. Lin et al. (2018) created a hybrid technique by

merging the DBN with a derivative-free optimization technique namely negative-correlation search (NCS) to optimize the hyperparameters of DBN to forecast the equity price and obtained efficient results as compared to traditional neural networks.

However, in the last few years, the long short-term memory (LSTM) neural network is an extensively accepted method for time series forecasting because it can process highly nonlinear and uncertain data. LSTM is a class of recurrent neural networks with memory cells and is significantly used for solving time series forecasting problems. Wang et al. (2018) used the LSTM network to forecast the Chinese stock market and attempted to improve the traditional gradient descent algorithms to enhance the learning process in LSTM. Fischer and Krauss (2018) deployed the LSTM model to predict the trend in various stocks of the S&P 500 index and compared the forecasting accuracy with three memory-free models like random forest (RF), deep neural network (DNN), and logistic regression (LR) and demonstrated the superior performance of the LSTM. Pang et al. (2020) presented the concept of a stock vector and developed two hybrid models: deep LSTM network with an embedded layer and LSTM with the automatic encoder to vectorize the input data that is fed as input to LSTM to achieve better forecasting results. Wei Bao et al. (2017) presented a multiple-stage novel deep learning approach for forecasting stock price. In the first stage, the stock price time series has been filtered by using wavelet transform (WT) to remove the noise, secondly, high-level features are extracted using stacked autoencoder (SAE) and finally, LSTM is applied to forecast the 1-day ahead close price. The authors showed that the proposed model performs significantly better than other approaches such as RNN, LSTM, and WT-LSTM.

Although the LSTM shows good performance in time series forecasting problems, the biggest challenge in LSTM neural network is tuning its hyperparameters namely number of hidden layers, number of hidden nodes in each layer, batch size, number of epochs, learning rate, and optimizing the connection weights as well bias of the network. The automatic search of the architecture of neural networks has been attempted by numerous authors (Elsken et al. 2019). Sakshi and Kumar (2019) used genetic algorithms to evolve the parameters of an ANN and found that the suggested model has a short training period, rapid convergence, and a greater success rate. Peng et al. (2018) applied differential evolution (DE) to obtain the optimal value of various hyperparameters such as window length, the number of hidden nodes, batch size, and the number of epochs in LSTM for electricity price prediction. Liu and Liu (2019) incorporate modified GA for selecting the optimal features subset and hidden neurons of LSTM neural networks for house price prediction in China.

In this study, one of the challenging problems considered is optimizing the initial parameters (weights and bias) of LSTM and fully connected layer (FCL) in a deep neural network (DNN) model for stock price forecasting. Gradient descent (GD) (Ruder 2016), stochastic gradient descent (SGD) (Schaul et al. 2013), stochastic gradient descent with momentum (SGDM) (Yazan and Talu 2017), root-mean-square propagation (RMSProp) (Hinton et al. 2012) and adaptive moment estimation (Adam) (Kingma and Ba 2014) are most popular used algorithms for training LSTM neural network. Among the learning algorithms, Adam optimizer is the most frequently used optimization method for training ANNs because it combines the advantages of two optimization techniques viz. SGDM and RMSProp. Despite being the good performance of Adam optimizer, it has two limitations: firstly, if the objective function is non-differentiable or multi-modal, it might become stuck in local minima (Bock and Weiß 2019). Secondly, it suffers from the problem of vanishing/exploding gradient due to the use of the sigmoid activation function in LSTM (Roodschild et al. 2020). This weakness of the Adam optimizer and the ability of nature-inspired optimization methods to handle continuous optimization problems, have motivated us to use them to find out the optimal set of initial weights for DNN.

Because of Adam's inability to converge locally, it's clear that ANN weight and bias' optimization are heavily reliant on the initial values (Roodschild et al. 2020). If the starting values are situated in the local search space then the networks may become stuck in the local region. This issue can be solved by using global search strategies to train ANNs. Initially, to reduce the search space, PSO is used to find the optimal values of initial weight and bias of the LSTM network and fully connected layer in the DNN model, and the resulting weights and bias are then used as the LSTM neural network's starting parameters.

PSO is a swarm intelligence-based stochastic optimization method inspired by the social behavior of animals. In PSO, the inertia coefficient maintains the balance among the exploration and exploitation properties and avoids premature convergence (Nickabadi et al. 2011). Eberhart and Shi (2000) introduced a time-varying inertia coefficient to enhance the exploration and exploitation features of PSO and obtained efficient results. Panigrahi et al. (2008) presented an adaptive PSO in which distinct inertia coefficients are given to particles based on their rank. In this study, we proposed an Adaptive PSO by improving the inertia coefficient of standard PSO to avoid the problem of being stuck in the local optimum during iteration. We have assigned the inertia coefficient to each particle during iteration based on the velocity of global best and personal best particles in the swarm.

In this study, we proposed a swarm intelligence-based hybrid deep learning approach for short-term (1-day, 1-week and 2-weeks ahead) and long-term (4-weeks, 6-weeks and 12 weeks ahead) forecasting of the price of equity market indices by applying technical indicators. The proposed technique is obtained by merging the Adaptive PSO and the Adam optimizer for training the LSTM neural network. Firstly, we employ the adaptive PSO for determining the initial weights and bias of LSTM units and fully connected layer (FCL) in the DNN model. Finally, the LSTM and FCL are initialized with the weights and bias obtained by adaptive PSO and the configured model is further trained by the Adam optimizer. The proposed model is named as Adaptive PSO-LSTM. This article also compares the model's predicting performance to that of conventional LSTM, Elman neural network (ENN) (Ren et al. 2018), and a hybrid approach created by incorporating evolutionary techniques like GA with LSTM. The proposed method's forecasting performance is tested by predicting the close price of three stock market indices: S&P 500, Sensex, and Nifty 50.

The following are the key contributions of this work:

- (1) The only data available in the stock market is the daily open, high, low, and close prices, as well as the number of shares traded, we developed a pool of technical indicators.
- (2) To handle the premature convergence issue of the standard PSO we proposed an Adaptive PSO to improve the inertia coefficient to enhance the global search capability of the particles and to prevent the particles from being trapped in local optimum and to balance the exploration and exploitation properties of PSO.
- (3) To deal with the issue of overfitting/underfitting and vanishing/exploding gradient in the LSTM neural network we have applied the Adaptive PSO to obtain the optimal value of the initial input weights, recurrent weights and bias of LSTM and input weights and bias of FCL in DNN model to achieve the high forecasting accuracy in the domain of stock price forecasting. To the best of our knowledge, this is the first study in which PSO is used to determine the initial weights and bias of LSTM and FCL in DNN.

The remainder of this article is structured as: Sect. 2 provides the outline of previous studies. Section 3 describes the framework for developing the proposed technique. The evaluation metrics and experimental setting are described in Sect. 4. The experimental findings and analyses are shown in Sect. 5. The conclusion is presented in Sect. 6.

## 2 Related work

Stock price time series forecasting is a challenging area of research due to various unknown factors that influence the stock price movement. In the past several years, machine learning (Ulke et al. 2018; Sands et al. 2015; Kumar et al. 2020) and deep learning (Livieris et al. 2020; Long et al. 2020; Nabipour et al. 2020; Ding and Qin 2020; Chong et al. 2017; Baldominos et al. 2020), metaheuristic optimization techniques (Mahajan et al. 2022c, d)-based approaches have gained popularity in time series forecasting and optimization domain. Kumar et al. (2021a) presented a systematic review of computational intelligence-based approaches for stock price forecasting and analyzed that the nature-inspired-based hybrid neural network models are extensively accepted approaches. Li and Bastos (2020) conducted a comprehensive survey that focuses on deep learning and technical analysis-based model for stock price forecasting and showed that LSTM neural network-based hybrid models are the most frequently applied approaches in the time series forecasting domain. Van Houdt et al. (2020) presented the review of the application of LSTM neural networks in various domains including time series prediction and discussed that the LSTM model is well suited for processing temporal sequences and is suitable for dealing with time series prediction problems. A comprehensive review of a hybrid model based on nature-inspired optimization for adjusting the hyper-parameters and topology of deep neural networks was recently presented by Darwish et al. (2020). In the area of data analytics, they stated that hybrid ANN models are adaptable and effective.

In very recent work, Jamous et al. (2021) developed a novel PSO called PSO with center of gravity (PSOCoG) to choose the optimal hyperparameters of ANN and evaluated the proposed model for stock price forecasting under the effect of the COVID-19 pandemic. Kamalov (2020) employed two deep neural network models such as CNN and LSTM, one shallow neural network viz. multilayer perceptron (MLP), and random forest (RF) to forecast the significant change in the close price of the stock market index. Experimental results confirm that the deep learning models achieved a high degree of accuracy as compared to machine learning models under consideration. Ji et al. (2021) designed a novel improved PSO and combined it with LSTM to optimize the hyperparameters of LSTM to create a hybrid DNN model for forecasting stock price. They attempted to improve the inertia weight in PSO by utilizing the nonlinear approach and then applied the improved PSO (IPSO) to LSTM to choose the optimum value of the number of epochs, learning rate, and the number of nodes in two hidden layers. Peng et al. (2020) applied the fruit fly optimization algorithm (FOA) to select

the optimal hyper-parameters of LSTM for time series forecasting. The hyper-parameters considered in this study are window length of time series, batch size, number of hidden neurons, and number of training iterations. Huang et al. (2021) proposed variational mode decomposition (VMD), LSTM, GA, and back-propagation neural network (BPNN)-based hybrid approach in the domain of financial time series forecasting. Firstly, they employed the VMD to decompose the time series into short-term and long-term trends and applied GA to tune the parameters of VMD to minimize the loss and then fed the decomposed data as input to LSTM. Liu and Long (2020) developed a hybrid deep learning framework by combining the empirical wavelet transform (EWT) and outlier robust extreme learning machine (ORELM) for preprocessing and post-processing of financial time series, LSTM as forecasting model, dropout strategy for optimizing the training process and PSO is used for determining the optimum hyperparameters such as number of hidden neurons, number of training epochs and learning rate of LSTM. Chung and Shin (2018) integrated GA with LSTM network to develop a hybrid deep learning approach to determine the optimal window length and optimal architecture of LSTM for stock market prediction and prove that the proposed model outperforms the benchmark models.

Kumar et al. (2021d) integrated sentimental analysis and technical analysis and developed a hybrid model by merging ABC with LSTM for forecasting time series data. The ABC is employed to tune the six parameters of LSTM namely window length, learning rate, number of hidden units, number of epochs, batch size, and dropout rate, and obtained enhanced forecasting accuracy. Hu et al. (2020) have proposed a hybrid model that predicts two major stock indexes, by merging extreme learning machine (ELM) with Improved Harris's hawks optimization (IHHO). In this paper, IHHO is utilized to find the best value for weights and bias for ELM. Sahoo and Mohanty (2020) used a hybrid technique that included ANN and the grey wolf optimization approach (GWO). The authors of this paper used GWO to fine-tune the parameters of an ANN for forecasting stock prices over different periods. Kumar et al. (2021c) presented a hybrid evolutionary ANN model by aggregating principal component analysis (PCA), PSO, and feed-forward neural network (FFNN) to foresee the close price of stock indices by employing 20 indicators. In this work, PCA is utilized for dimensionality reduction; PSO is applied to determine the optimum value of initial weights and bias of FFNN and obtained promising results.

ANN and nature-inspired algorithm (NIA)-based hybrid models have proven to be potential techniques for stock price time series forecasting. From the literature, it is evident that most of the researchers have attempted to optimize the various hyperparameters of LSTM such as



time window length, number of hidden layers, batch size, number of hidden neurons, learning rate, dropout rate, and number of epochs.. To the best of our knowledge, none of the studies has considered the optimization of initial weights and bias of LSTM neural network. Hence, we attempt to optimize the initial weights and bias of LSTM and FCL in the DNN model using adaptive PSO for stock price forecasting.

### 3 Methodology

To predict the stock price this article presents a deep learning framework comprised of four phases to develop PSO-based hybrid deep learning model. The first step consists of calculating technical indicators from daily stock prices and data pre-processing, which includes using the min-max normalization approach to map technical indicators to a narrow range. PSO is utilized in the second step to obtain the initial weights and bias of the LSTM and FCL in DNN. In the following step, we allocate the optimal weight and biases acquired in the preceding phase to the DNN and train the LSTM with the Adam optimizer. In the final phase, we analyze the forecasting efficiency of the model based on five performance measures and then yield the forecasted prices. The general framework of the proposed technique is depicted in Fig. 1. In the subsequent sections, each phase is described in detail.

#### A. Technical indicators and pre-processing

Initially, a pool of technical indicators is created using daily OHLC stock market data, which includes the open (OP) price, high (HI) price, low (LO) price, and closing (CL) price, as well as the volume (V). The technical indicators considered in this study are selected from prior research by Kumar et al. (2021b, c). Table 1 provides an overview of mathematical formulations of technical indicators.

#### B. Data normalization

To convert the data values in a short-range we use the Min-Max normalization technique. Let us assume that  $X_{min}$  and  $X_{max}$  are the lowest and highest values of a feature . Then this method map a value  $x$  of  $X$  to  $x^*$  from  $[X_{min}, X_{max}]$  to a small range  $[X_{min}^*, X_{max}^*]$  by Eq. (1) given as:

$$x^* = \left( \frac{x - X_{min}}{X_{max} - X_{min}} \right) (X_{max}^* - X_{min}^*) + X_{min}^* \quad (1)$$

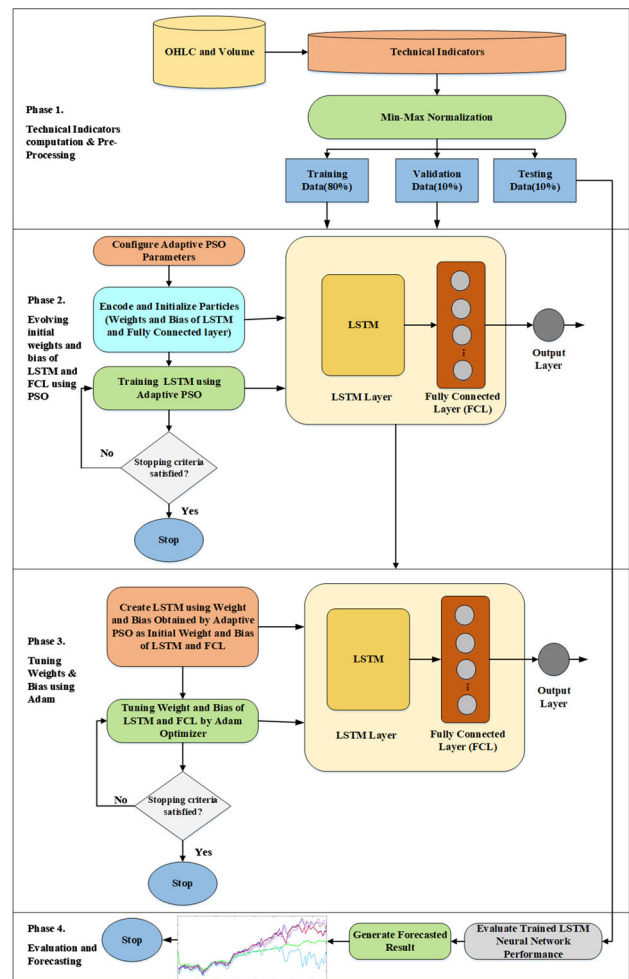


Fig. 1 The general framework of proposed work

#### C. Long short-term memory (LSTM)

The LSTM neural network developed by Hochreiter and Schmidhuber (1997a) is a type of recurrent neural network (RNNs) with nonlinear gated units and memory cells that enable it to capture long-term dependencies in sequential data (Greff et al. 2016). The LSTM has the capability to deal with vanishing/exploding gradient problems that typically occur in RNNs that restrict it to capture long-term dependencies in time series data (Hochreiter and Schmidhuber 1997b). The LSTM overcomes this problem by selectively reading, writing, and forgetting the information so that only important information remains in the memory cells.

The LSTM network is made-up of the memory cell and three gates namely input gate, forget gate, and output gate. The three gates manage the transfer of information through the memory cell by deciding which information to forget, retain and pass (output) to the next state. Figure 2 displays the typical structure of an LSTM network with a single

**Table 1** Technical indicators

Technical indicator	Formulation
Open price (OP <sub><i>t</i></sub> )	-
High price (HI <sub><i>t</i></sub> )	-
Low price (LO <sub><i>t</i></sub> )	-
Close price (CL <sub><i>t</i></sub> )	-
Simple moving averages (SMA <sub><i>t</i></sub> ( <i>n</i> ))	$SMA_t(n) = \frac{1}{n} \sum_{i=0}^{n-1} CL_{t-i}$
Exponential moving averages (EMA <sub><i>t</i></sub> ( <i>n</i> ))	$EMA_t(n) = \alpha(CL_t - EMA_{t-1}) + EMA_{t-1}$ , where $\alpha = \frac{2}{n+1}$ is smoothing factor
Moving averages convergence divergence (MACD <sub><i>t</i></sub> ( <i>n</i> , <i>k</i> ))	MACD <sub><i>t</i></sub> = EMA <sub><i>t</i></sub> ( <i>n</i> ) - EMA <sub><i>t</i></sub> ( <i>k</i> )
Average directional index (ADX <sub><i>t</i></sub> ( <i>n</i> ))	$ADX_t = \frac{ DI^+ - DI^- }{ DI^+ + DI^- } * 100$ , where, $DI^+ = EMA(DM^+, n) / ATR(n)$ is 'positive directional indicator' and $DI^- = EMA(DM^-, n) / ATR(n)$ is 'negative directional indicators' and $DM^+ = (HI_t - HI_{t-1})$ and $DM^- = (LO_{t-1} - LO_t)$ are directional movements
Commodity channel index (CCI <sub><i>t</i></sub> ( <i>n</i> ))	$CCI_t(n) = \frac{TP - SMA_t(TP, n)}{0.015 * \sum_{i=1}^n  TP_{t-i} - SMA_t(TP, n)  / n}$ where, TP = (CL <sub><i>t</i></sub> + HI <sub><i>t</i></sub> + LO <sub><i>t</i></sub> ) / 3 is typical price
Momentum (MTM <sub><i>t</i></sub> ( <i>n</i> ))	$MTM_t(n) = CL_t - CL_{t-n}$
Rate of change (ROC <sub><i>t</i></sub> ( <i>n</i> ))	$ROC_t(n) = \frac{CL_t - CL_{t-n}}{CL_{t-n}} * 100$
Relative strength index (RSI <sub><i>t</i></sub> ( <i>n</i> ))	$RSI_t(n) = 100 - \frac{100}{1 + \frac{\sum_{i=0}^{n-1} UP_{t-i}/n}{\sum_{i=0}^{n-1} DN_{t-i}/n}}$ Where, UP <sub><i>t</i></sub> is profit and DN <sub><i>t</i></sub> is loss at time <i>t</i>
True strength index (TSI <sub><i>t</i></sub> ( <i>n</i> , <i>k</i> ))	$TSI_t(n) = \frac{EMA(EMA(PC, n), k)}{EMA(EMA(PC, n), k)}$ Where, PC <sub><i>t</i></sub> = CL <sub><i>t</i></sub> - CL <sub><i>t-1</i></sub> is variation in close price
Stochastic %K (%K <sub><i>t</i></sub> ( <i>n</i> ))	$\%K_t(n) = \frac{CL_t - LL_{t-(n-1)}}{HH_{t-(n-1)} - LL_{t-(n-1)}}$
Stochastic %D (%D <sub><i>t</i></sub> ( <i>n</i> ))	$\%D_t(n) = \frac{\sum_{i=0}^{n-1} \%K_{t-i}}{n}$
William %R (%R <sub><i>t</i></sub> ( <i>n</i> ))	$\%R_t(n) = \frac{HH_{t-(n-1)} - CL_t}{HH_{t-(n-1)} - LL_{t-(n-1)}}$ where, HH <sub><i>t-n</i></sub> and LL <sub><i>t-n</i></sub> are the highest high price and lowest low price in last <i>n</i> days
Average true range (ATR <sub><i>t</i></sub> ( <i>n</i> ))	$ATR_t(n) = \frac{ATR_{t-1} * (n-1) + TR_t}{n}$ where, $TR_t = \max( HI_t - LO_t ,  HI_t - CL_{t-1} ,  LO_t - CL_{t-1} )$ is 'true range'
Ulcer index (UI <sub><i>t</i></sub> ( <i>n</i> ))	$UI_t(n) = \sqrt{\sum_{i=0}^{n-1} (R_i^+)^2}$ where, $R_i^+ = (CL_t - \max(CL_{t-n})) / \max(CL_{t-n})$ and $\max(CL_{t-n})$ is the maximum close price in previous <i>n</i> days
Accumulation distribution index (ADI <sub><i>t</i></sub> )	$ADI_t = ADI_{t-1} + CMFV$ where, $CMFV = \left( \frac{CL_t - LO_t}{(HI_t - LO_t)} - \frac{(HI_t - CL_t)}{(HI_t - LO_t)} \right) * V_t$ is 'current money flow value'
On-balance volume (OBV <sub><i>t</i></sub> )	$OBV_t = OBV_{t-1} + \begin{cases} V_t & \text{if } CL_t > CL_{t-1} \\ 0 & \text{if } CL_t = CL_{t-1} \\ -V_t & \text{if } CL_t < CL_{t-1} \end{cases}$ where, <i>V<sub>t</sub></i> is the number of 'shares traded on a trading day <i>t</i> '

*n* denotes a 10-days period, whereas *k* > *n* denotes a 15-days period

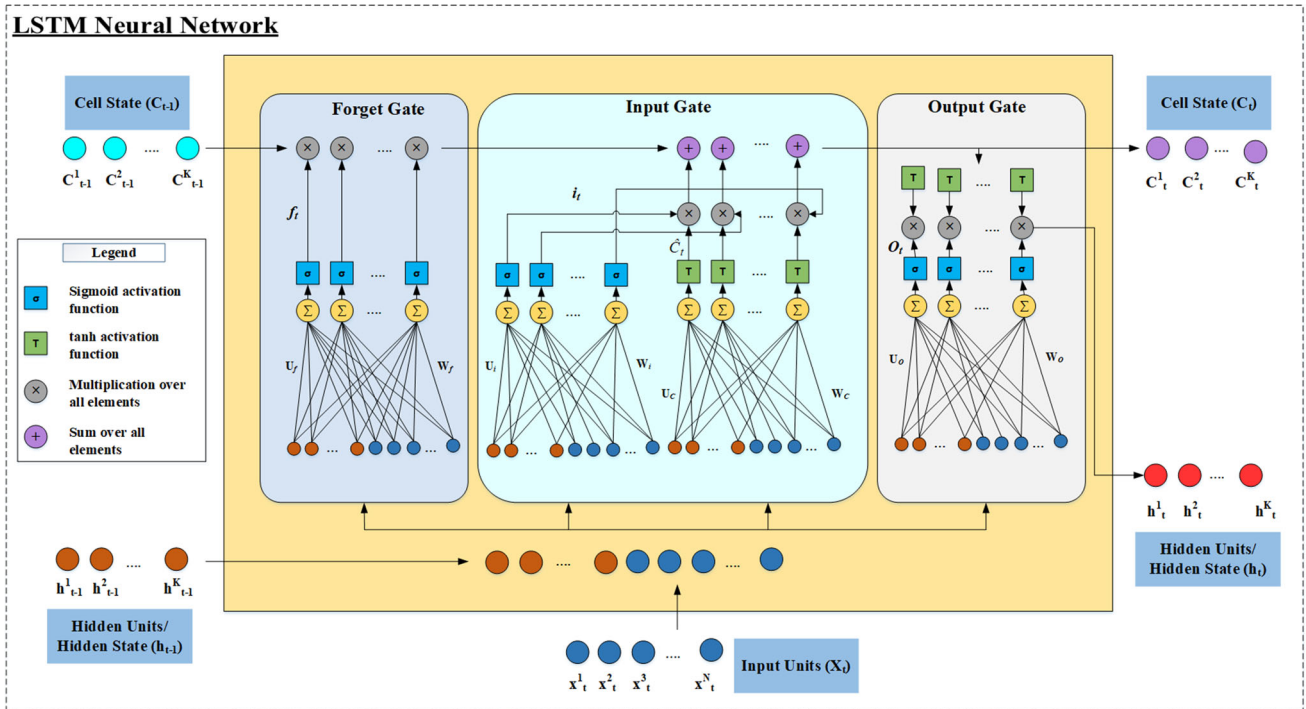


Fig. 2 Long short-term memory (LSTM) neural network

hidden layer. The core part of the LSTM unit contained in the hidden layer(s) is the cell state.

Let  $X_t = [x_t^1, x_t^2, x_t^3, \dots, x_t^N]$  be the  $N$  number of inputs,  $h_t = [h_t^1, h_t^2, h_t^3, \dots, h_t^K]$  be the  $K$  hidden units or hidden states and  $C_t = [c_t^1, c_t^2, c_t^3, \dots, c_t^K]$  be the cell state of the LSTM network at time  $t$  and  $f_t$ ,  $i_t$  and  $O_t$  represent the forget gate, input gate, and output gate, respectively. At each timestamp  $t$ , the input  $X_t$  along with previously hidden state  $h_{t-1}$  is presented to three gates to compute the next hidden state  $h_t$  and to update the previous cell state  $C_{t-1}$  in order to compute the new cell state  $C_t$ . The mathematical formulation of various operations performed in LSTM is given below:

- (1) In the first stage, the forget gate  $f_t$  in the LSTM layer decides which information is to be discarded from the prior cell state  $C_{t-1}$ , which can be calculated as:
 
$$f_t = \sigma(W_f X_t + U_f h_{t-1} + b_f) \quad (2)$$
- (2) In the next stage, the LSTM units specify which information is to be included in the cell state  $C_t$ . This process involves two operations: first, candidate vector  $\hat{C}_t$  captures all the information from the last state  $h_{t-1}$ , and current input  $X_t$ . Second, the input gate  $i_t$  selectively read the information to be inserted in the cell state  $C_t$ . Both the operations are computed as:

$$\hat{C}_t = \tanh(W_c X_t + U_c h_{t-1} + b_c) \quad (3)$$

$$i_t = \sigma(W_i X_t + U_i h_{t-1} + b_i) \quad (4)$$

- (3) In the third step, a new cell state  $C_t$  is computed on the basis of results of previous steps as:

$$C_t = f_t \times C_{t-1} + i_t \times \hat{C}_t \quad (5)$$

- (4) In the final step, the output gate  $O_t$  determined how much information is to be passed to compute the output  $h_t$  at the next timestamp:

$$O_t = \sigma(W_o X_t + U_o h_{t-1} + b_o) \quad (6)$$

$$h_t = O_t \times \tanh(C_t) \quad (7)$$

Where  $W_f, W_i, W_c, W_o$  are the weights matrices corresponding to the input  $X_t$  and  $U_f, U_i, U_c, U_o$  are recurrent weights matrices associated with previously hidden state  $h_{t-1}$  and  $b_f, b_i, b_c$ , and  $b_o$  are the bias vectors for forget gate, input gate, candidate solution, and output gate, respectively.  $\sigma(x) = 1/(1 + e^{-x})$  is log-sigmoid activation function and  $\tanh(x) = (e^x - e^{-x})/(e^x + e^{-x})$  is the hyperbolic tangent activation function. The log-sigmoid activation function generates values in the range 0 to 1 describing how much of each value passes through. The value “0” specifies nothing can pass through and the value “1” specifies everything can pass through and the value between 0 and 1 determines what fraction of value passes through.

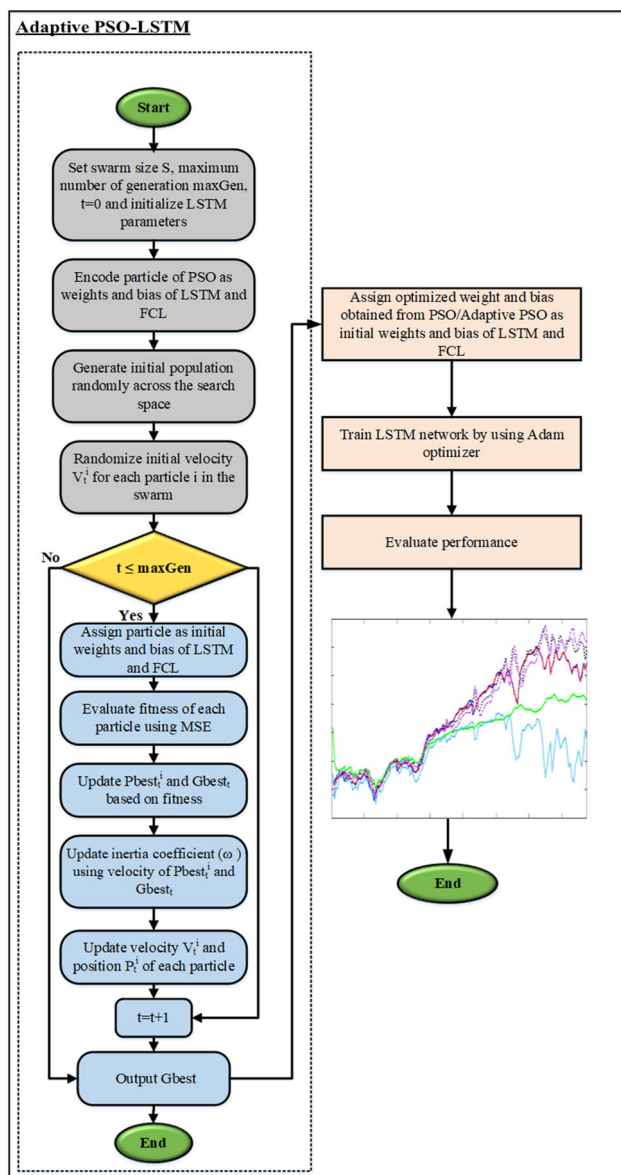


Fig. 3 Flowchart of adaptive PSO-based hybrid LSTM neural network

D. Proposed adaptive PSO-LSTM approach

In this work, our goal is to optimize the weights and bias of LSTM by applying Adaptive particle swarm optimization (PSO).

Kennedy and Eberhart (1995) introduced PSO, a population-based stochastic optimization approach inspired by the social behavior of fish schooling and bird flocks for addressing optimization problems in the continuous domain. The search for the best solution in PSO begins by assigning the population (swarm) with random solutions. In the D-dimensional search area, each particle represents a potential solution. Every particle in the swarm is evaluated and updated repeatedly during the search process. Because of its own best experience and the intelligence of the entire swarm, each particle travels to a new best position by adjusting its velocity.

When designing an LSTM, various hyper-parameters need to be considered and there are several ways to develop a DNN. In this study we considered a basic LSTM having one hidden layer connected to the input layer and a fully connected layer.

PSO is utilized to evolve the initial weights and biases of forget gate, input gate, candidate solution vector, and output gate of LSTM as well as weights and bias between fully connected layer and output layer. The main framework of the proposed adaptive PSO-based hybrid LSTM model is shown in Fig. 3. PSO comprises many phases, as illustrated in the diagram: encoding, initialization, evaluation, and update, which are explained as follows:

(1) Coding

In swarm intelligence-based methods, the most essential step is the representation of potential solutions (particles). Potential solutions are provided in this work by encoding the weights and biases of LSTM and FCL as swarm particles. The learnable weights in the LSTM network are input weights (IW), recurrent weights (RW), and bias (B). The matrices  $IW = [W_i, W_f, W_C, W_O]$ ,  $RW = [U_i, U_f, U_C, U_O]$ , and  $B = [b_i, b_f, b_C, b_O]$ , are the concatenation of input gate weights, forget gate weights, cell candidate weights, and output gate weights, respectively. The weights matrix “W” and bias “b” represent the parameters of FCL. Figure 4 shows how each particle (potential solution) in the PSO denotes the parameters in LSTM and FCL to be optimized.

(2) Initialization

In PSO, the dimensionality  $D$  of the search region is equivalent to the number of learnable parameters (weights and bias) in LSTM and FCL to be tuned given as:  $P = 4 * (K(K + N) + K)$ , where  $K$  is the number of hidden units and  $N$  is the total number of input variables and the

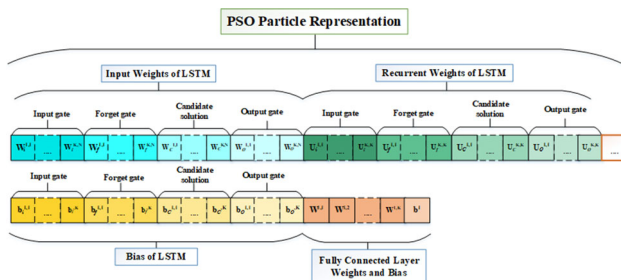


Fig. 4 PSO particle (solution) representation



number of parameters in FCL is equal to  $Q = O * K + O$ , where  $O$  is the number of nodes in the output layer. Thus, the total number of parameters to be optimized is equal to  $D = P + Q$ . The input weights ( $IW$ ) matrix in the particle is initialized with Glorot initializer (Glorot and Bengio 2010) which samples the weights from the uniform distribution with mean zero and variance  $2/(N + (4 * K))$  in the range  $[-\sqrt{6/(4 * K) + N}, \sqrt{6/(4 * K) + N}]$ . The recurrent weights ( $RW$ ) and weight matrices ( $W$ ) in the particle are initialized by the orthogonal matrix obtained by QR decomposition of the random matrix given by uniform normal distribution (Saxe et al. 2013). Finally, all the bias values are initialized with zero.

Initially, each particle's velocity  $V_0^i$  is assigned a zero value. Swarm or population size ( $S$ ), the maximum number of iterations (maxGen), personal coefficient ( $k_1$ ), social coefficient ( $k_2$ ), and inertia coefficient ( $\omega$ ), are various parameters of PSO that need to be appropriately set to execute it. The inertia coefficient  $\omega$  (Xiong et al. 2015), that varies from  $\omega_{max}$  to  $\omega_{min}$  with time in a linear manner is used in traditional PSO for comparison with adaptive PSO proposed in this study, which is defined as:

$$\omega = \omega_{max} - \frac{(\omega_{max} - \omega_{min})t}{T} \tag{8}$$

where  $t$  and  $T$  represent the current iteration and maximum iterations, respectively.

(3) Evaluation

The efficiency of every solution in the population is determined by calculating the cost function i.e., mean squared error between the real value and predicted value as described in Fig. 5. Given the data samples  $D = \{X_i, T_i\}_{i=1}^N$ , where,  $T_i$  is the target value corresponding to the input feature  $X_i$ . The cost function of the fitness function is given as follows:

$$f(IW, RW, B, W, b, D) = \frac{1}{N} \sum_{i=1}^N (T_i - Y_i)^2 \tag{9}$$

where  $N$  is the sample size and  $Y_i$  is the forecasted value of input  $X_i$  achieved from the output layer of LSTM and assigning the input weights ( $IW$ ), recurrent weights ( $RW$ ), and bias ( $B$ ) to LSTM and weights ( $W$ ) and bias ( $b$ ) to FCL generated by the PSO at iteration  $t$ .

(4) Update

The best solution obtained by each particle is known as personal best ( $Pbest$ ) and the optimum position occupied by any particle within the swarm known as global best ( $Gbest$ ), are improved at every iteration on the basis of fitness of each particle and following  $Pbest$  and  $Gbest$ , the

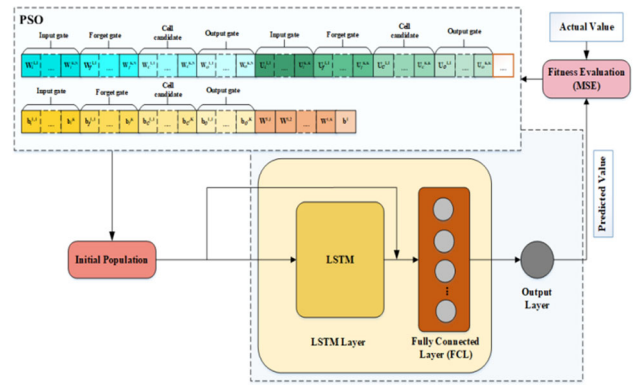


Fig. 5 PSO-LSTM iterative process for evaluating the fitness of network parameters of LSTM

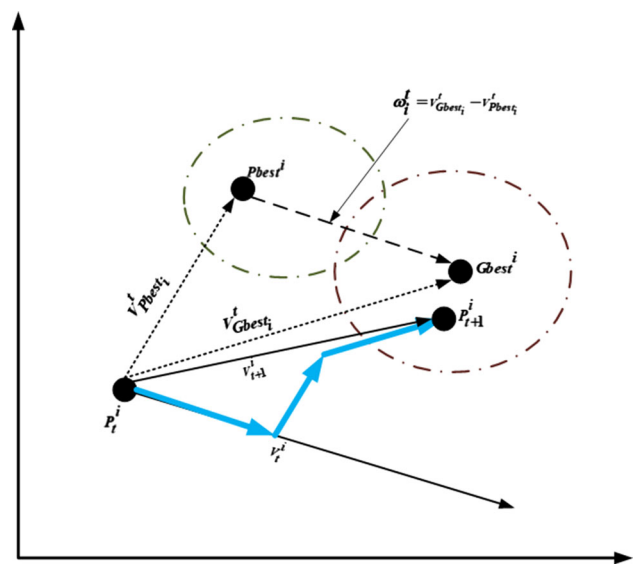


Fig. 6 Searching scheme of adaptive PSO

location and velocity of particles are modified during each iteration.

Let and  $V_t^i$  and  $P_t^i$  denote the velocity and position of  $i$ th particle at epoch  $t$ , respectively. Then the velocity  $V_{t+1}^i$  and the position  $P_{t+1}^i$  of  $i$ th particle at epoch  $t + 1$  are changed by Eqs. (10) and (11), respectively:

$$V_{t+1}^i = \omega \times V_t^i + k_1 r_1 \times (Pbest^i - P_t^i) + k_2 r_2 \times (Gbest^i - P_t^i) \tag{10}$$

$$P_{t+1}^i = P_t^i + V_{t+1}^i \tag{11}$$

where  $Pbest^i$  and  $Gbest^i$  represent the particle's personal the global best representing the potential solution, and  $r_1$  and  $r_2$  are two random numbers uniformly distributed in  $U(0,1)$ .

**Table 2** Description of stock price time-series data

Stock index	From	To	No. of samples	Training set	Validation set	Testing set
Nifty 50	Jan 1, 2015	March 31, 2021	1547	1237	155	155
Sensex	Jan 1, 2015	March 31, 2021	1541	1232	155	154
S&P 500	Jan 2, 2015	March 31, 2021	1572	1257	158	157

**Table 3** Parameters of the model

PSO		GA		LSTM	
Parameters	Values	Parameters	Values	Parameters	Values
Swarm size (S)	100	Population size	100	Topology (input-hidden-output)	20-100-1
$\omega_{\max}$	0.9	Crossover probability	0.7	Cell activation function	Tanh
$\omega_{\min}$	0.4	Mutation probability	0.1	Hidden state transfer function	Tanh
$k_1$	1.5	Gamma	0.4	Gate transfer function	Log-sigmoid
$k_2$	1.5	Selection pressure, beta	8	Epochs	100
maxGen	50	maxGen	50	Batch size	128
tolerance	$10^{-8}$	–	–	Learning rate, lr	0.01
–	–	–	–	GradientDecayFactor ( $\beta_1$ )	0.9
–	–	–	–	SquaredGradientDecayFactor( $\beta_2$ )	0.999

The inertia coefficient ( $\omega$ ) is one of the important parameters of PSO that maintain the balance amid exploitation and exploration features of PSO (Shi and Eberhart 1998). Exploitation is the property of the PSO to enhance the best solution it has achieved so far in the small neighborhood of the current best solution, whereas exploration is the characteristics of the PSO to search for new solutions by finding the regions with potentially better solutions. The large value of the inertia coefficient may result in divergence from the global best solution and a small value of the inertia coefficient may lead to slow convergence and high complexity. Therefore, the optimum value of the inertia coefficient is desired. In the literature, authors have attempted various strategies such as constant, time-varying, and adaptive inertia coefficients to maintain the balance between the exploitation and exploration properties of PSO (Nickabadi et al. 2011; Jamous et al. 2021; Ji et al. 2021). In this work, we use the adaptive mechanism for dynamically determining the value of the inertia coefficient by utilizing the fitness function (MSE) as a feedback parameter. To maintain the exploitation and exploration features of PSO we use the velocity of the personal best and global best particle in the swarm. The adaptive inertia coefficient is given as:

$$\omega_i^t = \omega_{\max} - (\omega_{\max} - \omega_{\min}) \times (V_{Gbest_i}^t - V_{Pbest_i}^t) \quad (12)$$

where  $\omega_i^t$  is the adaptive inertia coefficient,  $V_{Gbest_i}^t$  is the velocity of global best ( $Gbest_i^t$ ),  $V_{Pbest_i}^t$  is the velocity of personal best ( $Pbest_i^t$ ) particle  $i$  at iteration  $t$  and  $\omega_{\max}$  and

$\omega_{\min}$  are the maximum and minimum values of the inertia coefficient. Figure 6 shows the search mechanism of PSO using the adaptive inertia coefficient ( $\omega_i^t$ ) given by Eq. (12) in (10) in place of  $\omega$  to compute the velocity and position of particles using Eqs. (10) and (11), respectively.

The process of changing  $Pbest$ ,  $Gbest$ , location, and velocity of particle continue till the iteration count reaches maximum iterations (maxGen) or tolerance reaches a pre-set value. The final  $Gbest$  is the optimal solution comprised of tuned weights and bias of LSTM and FCL. Finally, the weights and bias generated by Adaptive PSO-LSTM are assigned as initial parameters of LSTM and FCL which is further trained by Adam optimizer.

## 4 Performance metrics and implementation of forecasting model

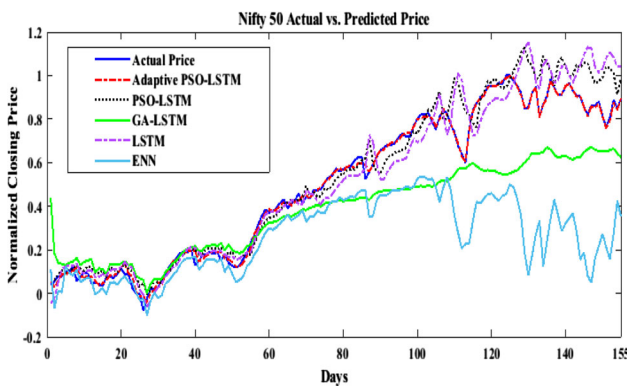
The performance metrics utilized to assess the proposed model's predictive abilities, as well as the experimental setup, are discussed in this section.

### A. Evaluation measures

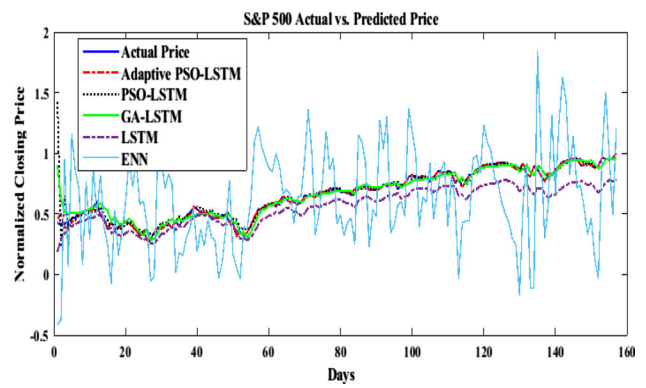
We employed five measures namely mean squared error (MSE), mean arctangent absolute percentage error (MAAPE), root-mean-square error (RMSE) (Botchkarev 2018; Kim and Kim 2016), symmetric mean absolute percentage error (SMAPE) (Hyndman and Koehler 2006) and Theil's Inequality Coefficient (Theil' U) (Theil 1966)

**Table 4** 1-Day ahead forecasting performance

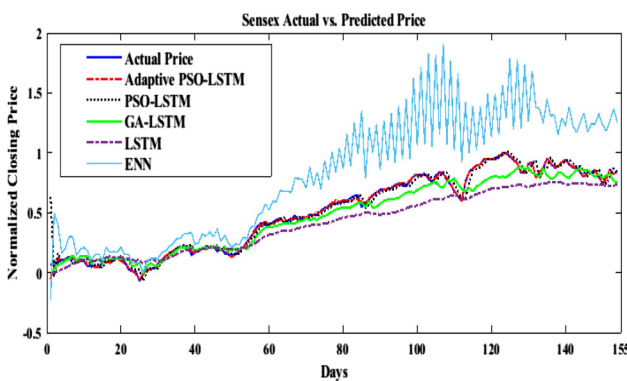
Stock Indices	Models	MSE	RMSE	MAAPE (%)	SMAPE (%)	Theil's U
Nifty50	ENN	1.15E-01	3.39E-01	43.32	16.49	3.71E-01
	LSTM	1.15E-02	1.07E-01	22.79	6.28	8.65E-02
	GA-LSTM	3.85E-02	1.96E-01	35.94	10.02	1.90E-01
	PSO-LSTM	8.20E-03	9.04E-02	20.74	5.30	7.23E-02
	Adaptive PSO-LSTM	2.98E-04	1.73E-02	10.17	2.77	1.44E-02
Sensex	ENN	1.54E-01	3.93E-01	57.65	13.23	2.51E-01
	LSTM	1.83E-02	1.35E-01	28.55	8.46	1.24E-01
	GA-LSTM	5.90E-03	7.65E-02	19.19	5.20	6.62E-02
	PSO-LSTM	3.60E-03	6.03E-02	14.09	3.99	5.00E-02
	Adaptive PSO	2.42E-04	1.56E-02	5.97	1.74	1.29E-02
S&P500	ENN	1.55E-01	3.93E-01	45.34	14.92	2.74E-01
	LSTM	1.36E-02	1.16E-01	14.91	4.15	9.19E-02
	GA-LSTM	3.20E-03	5.70E-02	6.09	1.46	4.13E-02
	PSO-LSTM	8.20E-03	9.06E-02	6.08	1.48	6.53E-02
	Adaptive PSO-LSTM	1.31E-04	1.15E-02	0.99	0.24	8.30E-03



**Fig. 7** 1-Day ahead forecasting results of Nifty 50



**Fig. 9** 1-Day ahead forecasting results of S&P 500



**Fig. 8** 1-Day ahead forecasting results of Sensex

to examine and compare the model's predicting accuracy and robustness. The following are the mathematical definitions of these metrics:

$$MSE = \frac{1}{N} \sum_{i=1}^N (A_i - P_i)^2 \tag{13}$$

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (A_i - P_i)^2} \tag{14}$$

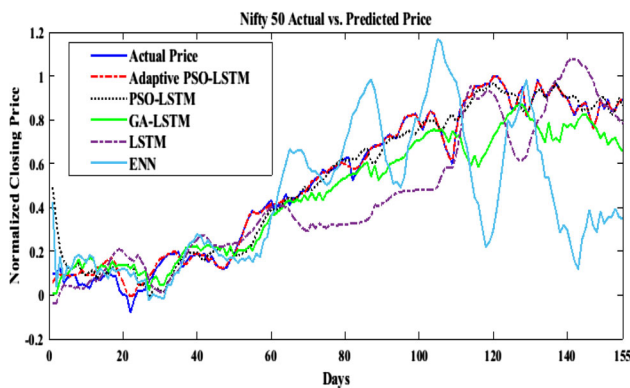
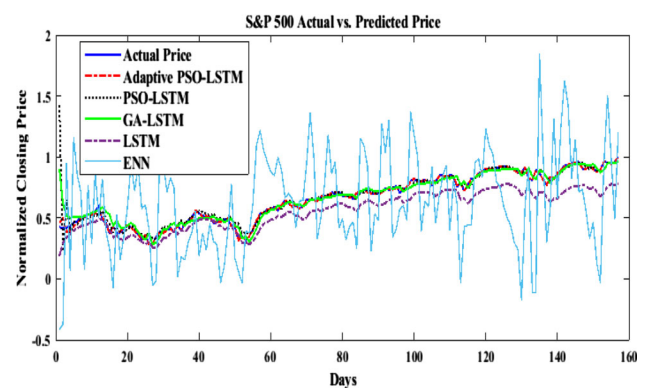
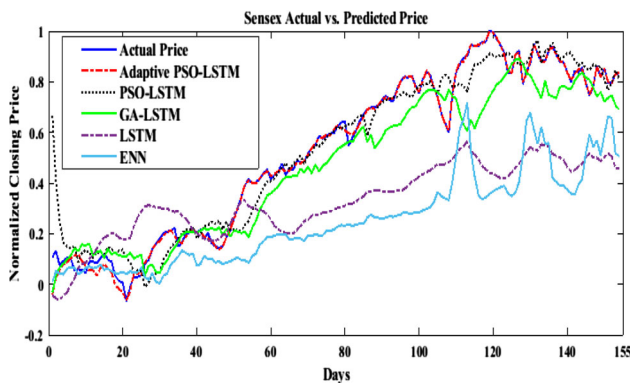
$$MAAPE = 100 \times \frac{1}{N} \sum_{i=1}^N \arctan \left( \left| \frac{A_i - P_i}{A_i} \right| \right) \tag{15}$$

$$SMAPE = 100 \times \frac{1}{N} \sum_{i=1}^N \left( \frac{|A_i - P_i|}{(|A_i| + |P_i|)/2} \right) \tag{16}$$

$$Theil' U = \frac{\sqrt{\frac{1}{N} \sum_{i=1}^N (A_i - P_i)^2}}{\sqrt{\frac{1}{N} \sum_{i=1}^N (A_i)^2 + \frac{1}{N} \sum_{i=1}^N (P_i)^2}} \tag{17}$$

**Table 5** 1-Week ahead forecasting performance

Stock Indices	Models	MSE	RMSE	MAAPE (%)	SMAPE (%)	Theil's U
Nifty50	ENN	8.48E-02	2.91E-01	44.55	14.35	2.53E-01
	LSTM	2.88E-02	1.69E-01	34.83	11.08	1.46E-01
	GA-LSTM	1.58E-02	1.26E-01	30.57	8.74	1.09E-01
	PSO-LSTM	5.20E-03	7.24E-02	24.08	6.92	5.86E-02
	Adaptive PSO-LSTM	4.63E-04	2.15E-02	13.36	3.61	1.74E-02
Sensex	ENN	1.11E-01	3.33E-01	49.95	19.19	3.59E-01
	LSTM	8.24E-02	2.87E-01	48.37	15.95	2.94E-01
	GA-LSTM	1.31E-02	1.14E-01	27.04	7.84	9.87E-02
	PSO-LSTM	7.20E-03	8.47E-02	23.31	6.58	6.87E-02
	Adaptive PSO	4.49E-04	2.12E-02	9.84	3.21	1.72E-02
S&P500	ENN	5.91E-01	7.69E-01	79.62	42.69	9.01E-01
	LSTM	1.08E-02	1.04E-01	15.83	4.12	8.41E-02
	GA-LSTM	8.90E-03	9.43E-02	13.12	3.48	1.41E-01
	PSO-LSTM	4.40E-03	6.61E-02	8.79	2.11	5.10E-02
	Adaptive PSO-LSTM	1.72E-04	1.31E-02	1.63	0.40	1.01E-02

**Fig. 10** 1-Week ahead forecasting results of Nifty 50**Fig. 12** 1-Week ahead forecasting results of S&P 500**Fig. 11** 1-Week ahead forecasting results of Sensex

where  $A_i$  is the true value and  $P_i$  is the predicted value derived from the proposed model for the  $i$ th observation of the dataset and  $N$  is the sample size in the dataset.

### B. Implementation of the proposed model

The forecasting models are implemented in MATLAB 2021a installed on a system with a 2.10 GHz i3 CPU and 4 GB RAM.

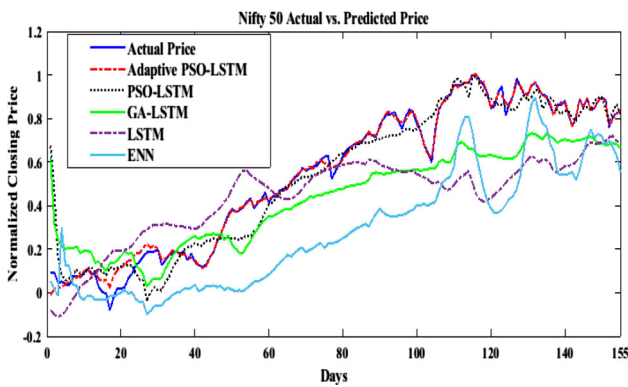
#### (1) Datasets

To show and evaluate the predicting performance of the model, we utilized stock price time-series data from three stock market indexes: S&P 500, Sensex, and Nifty 50 over the duration and number of samples presented in Table 2. The datasets used in this work were taken from Yahoo Finance. Each dataset is split into three parts: training data (80%), validation data (10%), and testing data (10%). The

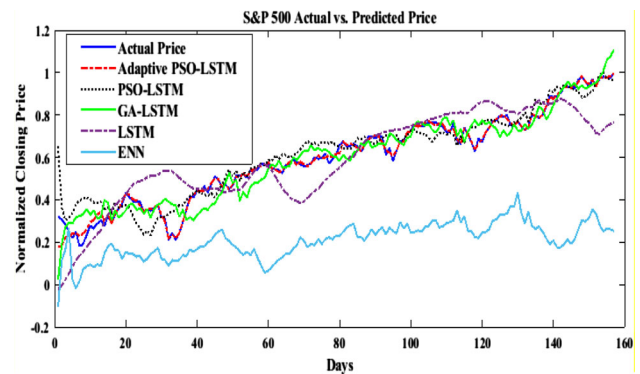


**Table 6** 2-Week ahead forecasting performance of the models

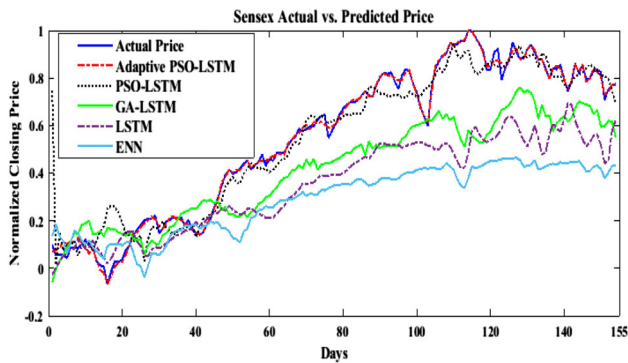
Stock Index	Model	MSE	RMSE	MAAPE (%)	SMAPE (%)	Theil's U
Nifty50	ENN	9.04E-02	3.11E-01	56.38	25.15	2.93E-01
	LSTM	5.19E-02	2.28E-01	44.49	13.35	2.04E-01
	GA-LSTM	3.46E-02	1.85E-01	41.30	11.76	1.66E-01
	PSO-LSTM	8.50E-03	9.25E-02	24.91	7.64	7.34E-02
	Adaptive PSO-LSTM	7.54E-04	2.75E-02	12.51	3.82	2.16E-02
Sensex	ENN	1.03E-01	3.20E-01	48.32	16.80	3.35E-01
	LSTM	5.83E-02	2.41E-01	39.38	12.82	2.33E-01
	GA-LSTM	3.68E-02	1.91E-01	36.87	10.98	1.75E-01
	PSO-LSTM	7.80E-03	8.84E-02	19.54	5.46	7.08E-02
	Adaptive PSO	2.87E-04	1.70E-02	7.75	2.27	1.34E-02
S&P500	ENN	1.02E-01	3.19E-01	42.91	15.31	3.27E-01
	LSTM	2.06E-02	1.43E-01	25.12	6.47	1.09E-01
	GA-LSTM	1.62E-02	1.27E-01	17.86	4.56	9.87E-02
	PSO-LSTM	7.77E-03	8.76E-02	11.52	2.77	7.06E-02
	Adaptive PSO-LSTM	7.43E-04	2.73E-02	3.89	0.93	2.20E-02



**Fig. 13** 2-Week ahead forecasting results of Nifty 50



**Fig. 15** 2-Week ahead forecasting results of S&P 500



**Fig. 14** 2-Week ahead forecasting results of Sensex

first 1237 data of Nifty 50 from January 1, 2015, to January 3, 2020, are considered as training dataset, next 155 data samples from January 6, 2020, to August 18, 2020, are used as validation dataset and the last 155 data from August 19, 2020, to March 31, 2021, are used as test dataset. The first 1232 data of Sensex from January 1, 2015, to January 6,

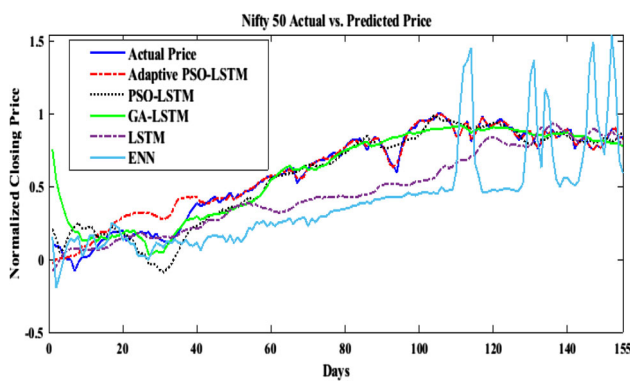
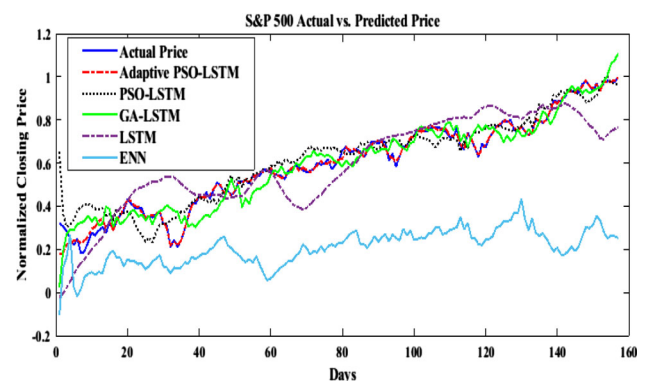
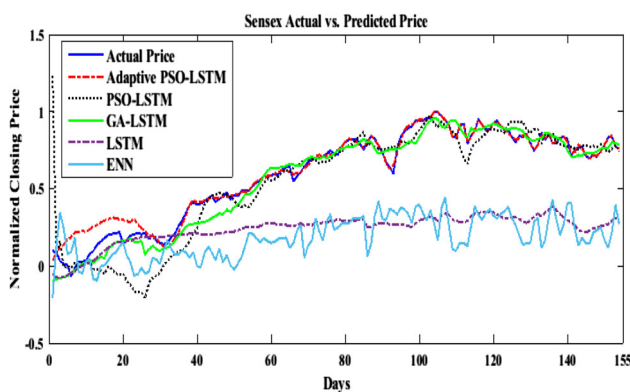
2020, are used as training dataset, the next 155 data from January 7, 2020, to August 18, 2020, are used as validation dataset and the last 154 data from August 19, 2020, to March 31, 2021, are used as test dataset. The first 1257 data of S&P 500 from January 2, 2015, to December 30, 2019, are used as training dataset, next 158 data from December 31, 2019, to August 13, 2020, are used as validation dataset and the last 157 data from August 14, 2020, to March 31, 2021, are used as test dataset.

(2) *Implementation of adaptive PSO-based LSTM neural network*

To build the pool of technical indicators, we used OHLC data and the number of shares traded in the first stage. Then the data pre-processing module is given a feature set of 20 indicators comprised of 16 technical indicators and 4 daily stock prices. The TA python library has been used to generate these indicators (Technical indicators 2020), and

**Table 7** 4-Week ahead forecasting performance

Stock indices	Models	MSE	RMSE	MAAPE (%)	SMAPE (%)	Theil's U
Nifty50	ENN	1.06E-01	3.25E-01	47.86	17.06	2.73E-01
	LSTM	3.74E-02	1.93E-01	29.79	9.60	1.60E-01
	GA-LSTM	1.11E-02	1.05E-01	21.14	6.15	7.91E-02
	PSO-LSTM	8.00E-03	8.93E-02	23.78	8.19	6.73E-02
	Adaptive PSO-LSTM	3.90E-03	6.23E-02	20.09	5.59	4.63E-02
Sensex	ENN	2.14E-01	4.62E-01	64.47	29.08	7.08E-01
	LSTM	1.77E-01	4.12E-01	52.14	20.63	6.18E-01
	GA-LSTM	4.48E-03	6.69E-02	17.60	5.91	5.11E-02
	PSO-LSTM	2.30E-02	1.19E-01	26.70	11.51	1.15E-01
	Adaptive PSO	3.30E-03	5.74E-02	15.29	4.02	4.33E-02
S&P500	ENN	1.78E-01	4.24E-01	56.89	24.23	4.93E-01
	LSTM	1.48E-02	1.22E-01	17.96	5.19	9.62E-02
	GA-LSTM	2.49E-02	1.58E-01	23.49	5.26	1.19E-01
	PSO-LSTM	5.00E-03	7.05E-02	11.98	2.84	5.49E-02
	Adaptive PSO-LSTM	4.18E-04	2.04E-02	2.68	0.71	1.61E-02

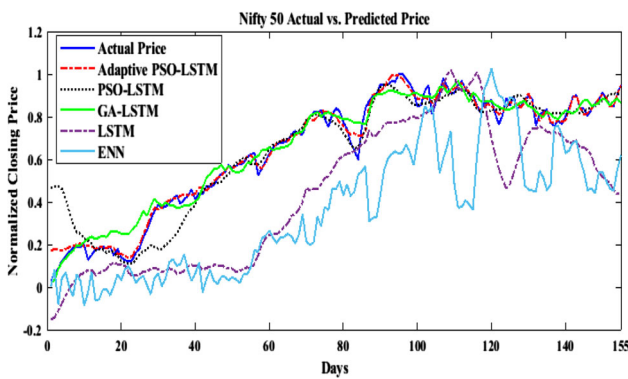
**Fig. 16** 4-Week ahead forecasting results Nifty 50**Fig. 18** 4-Week ahead forecasting results of S&P 500**Fig. 17** 4-Week ahead forecasting results of Sensex

then the Min–Max normalization approach is used to translate the data in the range  $[-1, 1]$ .

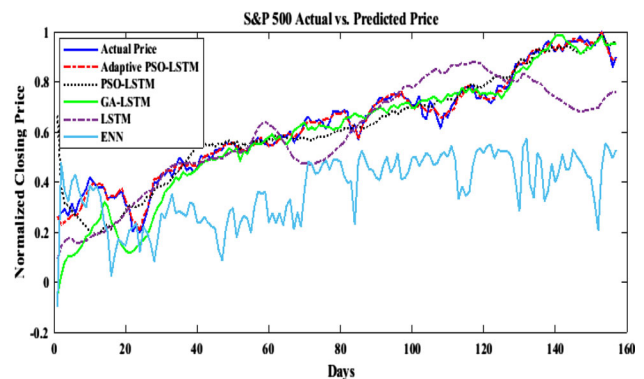
In the next phase, we implement adaptive PSO to improve the initial weights and biases for LSTM and FCL in DNN. The LSTM network considered in this study comprises an input layer with 20 nodes, 1 hidden layer with 100 neurons, and an output layer with 1 neuron. In the third phase, we employed Adam optimizer to train the LSTM created by assigning the initial weights and bias evolved in the preceding step. Table 3 shows the parameters required to implement the forecasting model. The value of the personal coefficient ( $k_1$ ) and social coefficient ( $k_2$ ) of PSO is set to be 1.2 (Ji et al. 2021). The value of maximum inertia coefficient, ( $\omega_{\max}$ ) and minimum inertia coefficient, ( $\omega_{\min}$ ) are set to be 0.9 and 0.4, respectively (Wang et al. 2020).

**Table 8** 6-Week ahead forecasting performance

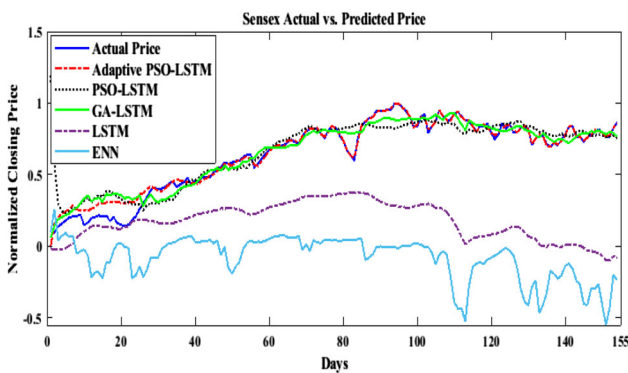
Stock indices	Models	MSE	RMSE	MAAPE (%)	SMAPE (%)	Theil's U
Nifty50	ENN	1.18E-01	3.45E-01	48.21	21.66	2.97E-01
	LSTM	7.01E-02	2.64E-01	39.19	16.18	2.13E-01
	GA-LSTM	2.90E-03	5.34E-02	10.81	2.53	3.78E-02
	PSO-LSTM	8.01E-03	8.98E-02	13.28	3.58	6.42E-02
	Adaptive PSO-LSTM	9.41E-04	3.07E-02	6.20	1.48	2.18E-02
Sensex	ENN	6.26E-01	7.91E-01	83.21	46.50	9.19E-01
	LSTM	2.84E-01	5.32E-01	58.23	27.91	5.86E-01
	GA-LSTM	5.10E-03	7.17E-02	13.90	3.20	5.18E-02
	PSO-LSTM	1.63E-02	1.28E-01	17.20	4.09	9.20E-02
	Adaptive PSO	1.80E-03	4.21E-01	9.27	2.25	3.04E-02
S&P500	ENN	9.31E-02	3.05E-01	38.69	13.79	2.90E-01
	LSTM	1.64E-02	1.28E-01	16.96	4.86	9.97E-02
	GA-LSTM	6.20E-03	9.89E-02	11.92	4.09	6.03E-02
	PSO-LSTM	4.80E-03	6.96E-02	9.82	2.64	5.28E-02
	Adaptive PSO-LSTM	3.19E-04	1.79E-02	2.65	0.67	1.35E-02



**Fig. 19** 6-Week ahead forecasting results of Nifty 50



**Fig. 21** 6-Week ahead forecasting results of S&P 500



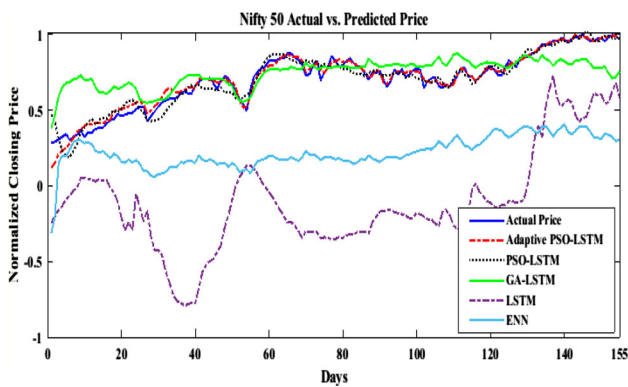
**Fig. 20** 6-Week ahead forecasting results of Sensex

## 5 Experimental results and discussion

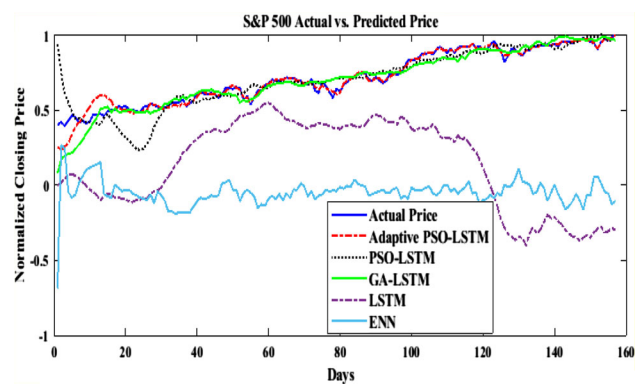
To evaluate the short-term (1-day, 1-week and 2-weeks) and long-term (4-weeks, 6-weeks and 12-weeks) forecasting efficiency of the proposed model entitled Adaptive PSO-LSTM, we have calculated five measures described in section IV and compared its forecasting efficiency with ENN, regular LSTM, GA-LSTM, and PSO-LSTM. Table 4 shows the experimental results corresponding to forecasting of 1-day ahead close price of three major stock market indices such as ‘Nifty 50, Sensex and S&P 500’. The ‘MSE, MAAPE, RMSE, SMAPE, and Theil’s U’ values of the Adaptive PSO-LSTM model for each case show that

**Table 9** 12-Week ahead forecasting performance of the models

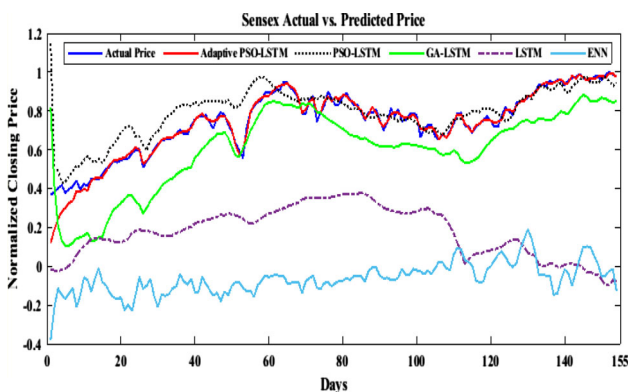
	MSE	Stock index	Model	SMAPE (%)	Theil's U
<i>Nifty50</i>					
ENN	2.57E-01	5.07E-01	58.45	25.94	5.19E-01
LSTM	7.45E-01	8.64E-01	83.32	43.29	8.00E-01
GA-LSTM	1.97E-02	1.40E-01	17.59	4.08	9.50E-02
PSO-LSTM	2.50E-03	4.98E-02	6.81	1.73	3.42E-02
Adaptive PSO-LSTM	1.11E-03	3.36E-02	4.44	1.19	2.30E-02
<i>Sensex</i>					
ENN	6.77E-01	8.29E-01	83.13	48.87	4.94E-02
LSTM	2.84E-01	5.32E-01	58.91	27.91	3.50E-03
GA-LSTM	2.84E-02	1.68E-01	21.89	7.04	2.00E-03
PSO-LSTM	1.04E-02	1.02E-01	10.32	2.41	2.67E-03
Adaptive PSO	1.30E-03	3.54E-02	3.12	0.92	8.96E-05
<i>S&amp;P500</i>					
ENN	6.28E-01	7.93E-01	81.31	48.33	9.43E-01
LSTM	4.80E-01	6.93E-01	61.60	30.11	6.50E-01
GA-LSTM	4.00E-03	6.34E-02	6.67	1.98	4.30E-02
PSO-LSTM	8.00E-03	8.97E-02	8.83	2.41	6.07E-02
Adaptive PSO-LSTM	1.43E-03	3.79E-02	3.55	0.94	2.56E-02



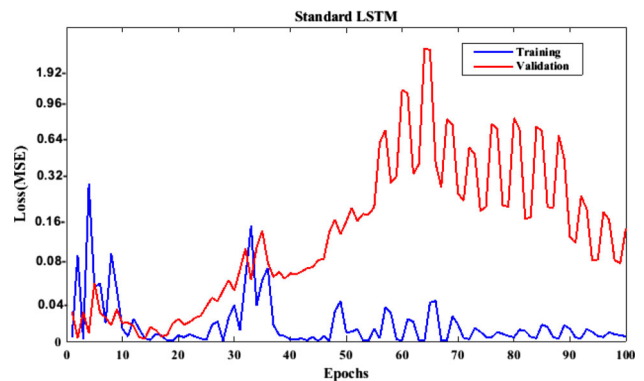
**Fig. 22** 12-Week ahead forecasting results of Nifty 50



**Fig. 24** 12-Week ahead forecasting results of S&P 500



**Fig. 23** 12-Week ahead forecasting results of Sensex



**Fig. 25** Training versus validation loss for standard LSTM

the model developed in this study performs better than other forecasting techniques considered in this study.

Furthermore, in each dataset, the performance measures values for LSTM are all higher than those for Adaptive PSO-LSTM, PSO-LSTM, and GA-LSTM. Therefore, we



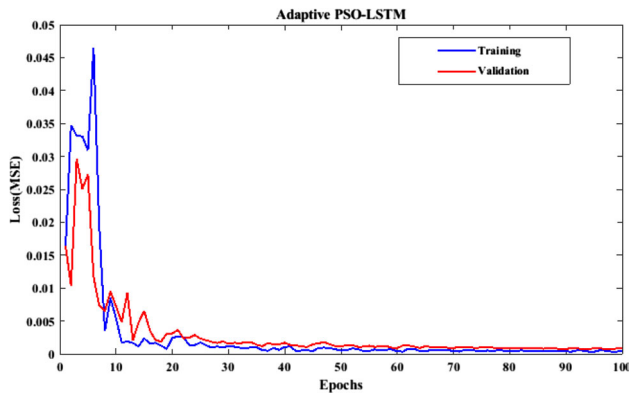


Fig. 26 Training versus validation loss for adaptive PSO-LSTM

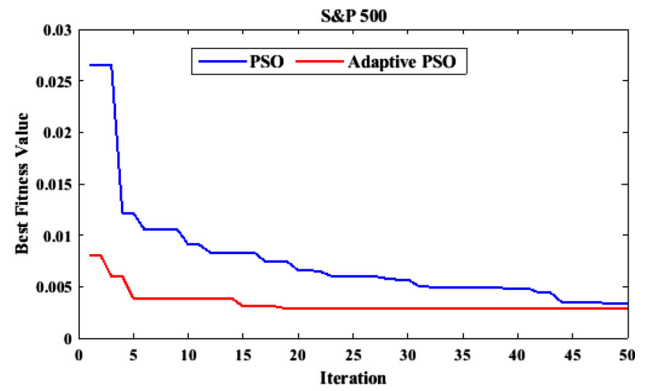


Fig. 29 Best fitness value (MSE) for S&P 500

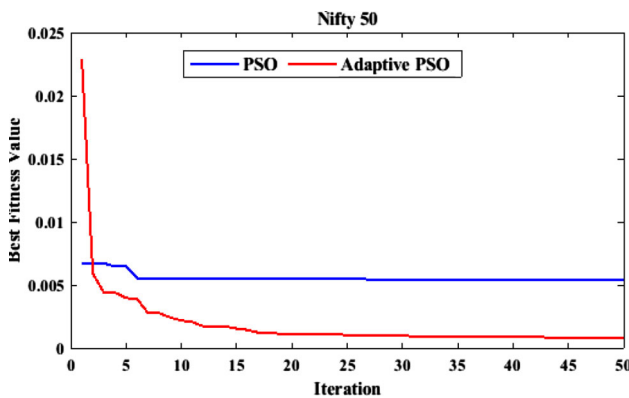


Fig. 27 Best fitness value (MSE) for Nifty 50

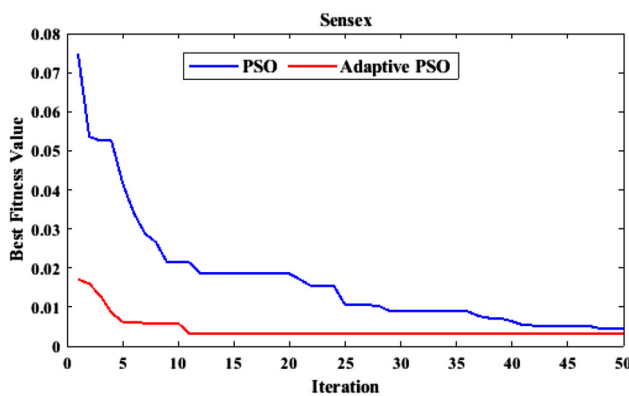


Fig. 28 Best fitness value (MSE) for Sensex

can infer that the proposed adaptive PSO has the capability to improve the parameters of a conventional LSTM network for time series forecasting.

The visualization of 1-day ahead forecasting results of three indices using test dataset by five models are shown by ‘Figs. 7, 8, and 9’. It is analyzed from the graphs of three indices that the true prices (blue line) are very close to the predicted price (red line) obtained by Adaptive PSO-LSTM

and hence the forecasting performance of the proposed model is higher as compared to others.

Table 5 shows the 1-week ahead forecasting results for three stock indices using five models, and Figs. 10, 11, and 12 show the plot of actual versus estimated price for 1-week ahead forecasting of three stock market indices. In the same way, we have concluded that the proposed technique performs better than others for 1-week ahead forecasting based on the findings.

Table 6 presents the performance metrics values obtained by applying the five models to forecast the 2-week ahead close price of three indices and Figs. 13, 14 and 15 show the closeness between the actual and predicted price for three datasets. It is clear from the plots that the price predicted by Adaptive PSO-LSTM is very close to actual price.

Table 7 presents the performance of the proposed Adaptive PSO-LSTM model in comparison to PSO-LSTM, GA-LSTM, LSTM, and ENN corresponding to the 4-week ahead forecasting results of three indices. It is worth mentioning that Adaptive PSO-LSTM gives the best forecasting performance in three datasets according to all performance metrics considered in this study.

The 4-week ahead forecasted value of the normalized closing price of three indices is portrayed in Figs. 16, 17, and 18, respectively. From the figures, we can judge that the close price forecasted by LSTM and ENN have huge differences compared to true price than GA-LSTM, PSO-LSTM, and Adaptive PSO-LSTM. Furthermore, the values predicted by Adaptive PSO are very close to the actual value which indicates the superiority of the proposed model in comparison to others.

As reported in Table 8, the proposed Adaptive PSO-LSTM model shows the better forecasting performance as compared to PSO-LSTM, GA-LSTM, LSTM, and ENN for 4-week ahead forecasting of ‘Nifty 50, Sensex, and S&P 500’. The actual price versus predicted price by different models is plotted in Figs. 19, 20, and 21 for three datasets

display that the proposed model performs better in comparison with others.

The comparison of 12-week ahead forecasting performance of proposed model Adaptive PSO-LSTM with PSO-LSTM, GA-LSTM, LSTM and ENN based on five metrics for three datasets is reported in Table 9. Our model demonstrates the best overall performance in terms of MSE, RMSE, MAAPE, SMAPE and Theil'U metrics. Furthermore, the graph of actual price versus predicted price shown in Figs. 22, 23 and 24 obtained by applying Adaptive PSO-LSTM and other models shows that the close price predicted by the proposed model is almost overlapping with the actual value which confirms the reliability of the model.

In general, the experimental findings obtained by implementing the proposed model on three datasets shows that the forecasted stock prices are very close to actual prices signifying the generality and robustness of the proposed Adaptive PSO-LSTM model for 'short term (1-day, 1-week and 2-weeks) and long term forecasting' (4-weeks, 6-weeks and 12-weeks).

Figures 25 and 26 show the graphical representation of training versus validation loss (MSE) with respect to the epochs for standard LSTM and Adaptive PSO-LSTM, respectively. It is evident from Fig. 25 that overfitting has occurred in standard LSTM because there is a large disparity between training and validation loss. However, the proposed model namely Adaptive PSO-LSTM has effectively avoided the overfitting issue because the training and testing errors are almost identical.

Furthermore to verify the robustness and reliability of Adaptive PSO-LSTM as compared to PSO-LSTM we have shown the plot of the best fitness value (MSE) versus iterations in Figs. 27, 28, and 29 for 'Nifty 50, Sensex, and S&P 500' datasets corresponding to 1-day ahead forecasting.

Our objective is to use an adjustable inertia weight in PSO to improve the exploration and exploitation features. Exploitation refers to the PSO's ability to improve the best solution it has obtained so far within a limited region of the current best solution, whereas exploration refers to the PSO's ability to identify new solutions by locating regions with possibly superior solutions. From Figs. 25, 26, and 27 it is clear how the fitness of particle (MSE) is further improved in each training epoch by Adaptive PSO as compared to PSO. Further, the robustness and reliability of Adaptive PSO-LSTM can be verified by comparing the MSE, RMSE, MAAPE, SMAPE, and Theil'U values with PSO-LSTM corresponding to 'short term (1-day, 1-week and 2-weeks) and long term (4-weeks, 6-weeks and 12-weeks) forecasting' results.

## 6 Conclusion

This work proposed a nature-inspired optimization-based hybrid deep learning approach named as Adaptive PSO-LSTM by proposing an adaptive PSO technique that search automatically for the initial input weights, recurrent weights, and bias of long short-term memory (LSTM) networks as well as weights and bias of fully connected layer (FCL) for forecasting stock price time series. We evaluated the forecasting ability of the proposed Adaptive PSO-LSTM model on 'three stock market indices' namely S&P 500, Sensex, and Nifty 50 for short term (1-day, 1-week and 2-weeks) and long term (4-weeks, 6-weeks and 12-weeks) forecasting. We compared the forecasting efficacy of the model with PSO-LSTM and GA-LSTM, LSTM, and ENN. The following is a summary of the major findings of this study:

- (1) The proposed hybrid model is effective in obtaining the optimum weights and bias of LSTM and FCL and improves the exploration and exploitation capability of PSO by using the adaptive inertia coefficient in PSO that increases the accuracy of the model.
- (2) The proposed technique successfully combines the global search potential of adaptive PSO with the local search ability of Adam optimizer, decreasing the chance of getting trapped in local minima and overcoming the underfitting/overfitting issues.
- (3) Experimental findings attained by employing the proposed method on three stock indices for 'short term (1-day, 1-week and 2-weeks) and long term (4-weeks, 6-weeks and 12-weeks) forecasting shows that the proposed Adaptive PSO-LSTM performs better in comparison to PSO-LSTM, GA-LSTM, LSTM, and ENN. Hence, the Adaptive PSO-LSTM is proved to be a promising approach for short-term and long-term forecasting of stock price time-series data.

In the future the work can further be extended by taking into account the additional hyperparameters of LSTM viz. learning rate, number of epochs, number of hidden layers and number of hidden neurons, batch size, window length of time series, dropout rate, etc. To improve the model's hyperparameters, more nature-inspired and evolutionary techniques would be investigated in the future in other financial time series forecasting domains.

**Authors contribution** GK was involved in conceptualization, writing—original draft preparation, software and visualization. UPS was involved in methodology, result validation and writing—reviewing and editing. SJ was involved in writing—reviewing and supervision.

**Funding** No funding was received to assist with the preparation of this manuscript.

**Data availability** The datasets analyzed during the current study are available in the <https://finance.yahoo.com/repository>.

## Declarations

**Conflict of interest** On behalf of the other two authors, I declare that we have no conflict of interest in any terms like financial support or non-financial from any organizations/institutions, data used in this paper, etc.

**Informed consent** This research work does not involve any human or animals participation.

## References

- Atsalakis GS, Valavanis KP (2009) Surveying stock market forecasting techniques—part II: soft computing methods. *Expert Syst Appl* 36(3):5932–5941
- Balaji AJ, Ram DH, Nair BB (2018) Applicability of deep learning models for stock price forecasting an empirical study on BANKEX data. *Procedia Comput Sci* 143:947–953
- Baldominos A, Saez Y, Isasi P (2020) On the automated, evolutionary design of neural networks: past, present, and future. *Neural Comput Appl* 32(2):519–545
- Bao W, Yue J, Rao Y (2017) A deep learning framework for financial time series using stacked autoencoders and long-short term memory. *PLoS ONE* 12(7):e0180944
- Berradi Z, Lazaar M (2019) Integration of principal component analysis and recurrent neural network to forecast the stock price of casablanca stock exchange. *Procedia Comput Sci* 148:55–61
- Bock S, Weiß M (2019, July) A proof of local convergence for the Adam optimizer. In: 2019 international joint conference on neural networks (IJCNN). IEEE, pp 1–8
- Botchkarev A (2018) Performance metrics (error measures) in machine learning regression, forecasting and prognostics: properties and typology. *arXiv preprint arXiv:1809.03006*.
- Cao J, Wang J (2019) Stock price forecasting model based on modified convolution neural network and financial time series analysis. *Int J Commun Syst* 32(12):e3987
- CDSL (Central depository services (India) limited), Retrieved from: <https://www.cdslindia.com/About/OurBusiness.aspx>, Accessed on 15 Oct 2021
- Chen C, Zhang P, Liu Y, Liu J (2020) Financial quantitative investment using convolutional neural network and deep learning technology. *Neurocomputing* 390:384–390
- Chong E, Han C, Park FC (2017) Deep learning networks for stock market analysis and prediction: methodology, data representations, and case studies. *Expert Syst Appl* 83:187–205
- Chung H, Shin KS (2018) Genetic algorithm-optimized long short-term memory network for stock market prediction. *Sustainability* 10(10):3765
- Darwish A, Hassanien AE, Das S (2020) A survey of swarm and evolutionary computing approaches for deep learning. *Artif Intell Rev* 53(3):1767–1812
- Ding G, Qin L (2020) Study on the prediction of stock price based on the associated network model of LSTM. *Int J Mach Learn Cybern* 11(6):1307–1317
- Eberhart RC, Shi Y (2000, July) Comparing inertia weights and constriction factors in particle swarm optimization. In: Proceedings of the 2000 congress on evolutionary computation. CEC00 (Cat. No. 00TH8512), vol 1. IEEE, pp 84–88
- Elsken T, Metzen JH, Hutter F (2019) Neural architecture search: a survey. *J Mach Learn Res* 20(1):1997–2017
- Fama EF (1970) Efficient capital markets: a review of theory and empirical work. *J Finance* 25:383
- Fischer T, Krauss C (2018) Deep learning with long short-term memory networks for financial market predictions. *Eur J Oper Res* 270(2):654–669
- Gao S, Zhou M, Wang Y, Cheng J, Yachi H, Wang J (2018) Dendritic neuron model with effective learning algorithms for classification, approximation, and prediction. *IEEE Trans Neural Netw Learn Syst* 30(2):601–614
- Glorot X, Bengio Y (2010, March) Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the thirteenth international conference on artificial intelligence and statistics. JMLR Workshop and Conference Proceedings. AISTATS, Sardinia, Italy, pp 249–256
- Gong Z, Chen H, Yuan B, Yao X (2018) Multiobjective learning in the model space for time series classification. *IEEE Trans Cybern* 49(3):918–932
- Greff K, Srivastava RK, Koutník J, Steunebrink BR, Schmidhuber J (2016) LSTM: a search space odyssey. *IEEE Trans Neural Netw Learn Syst* 28(10):2222–2232
- Guresen E, Kayakutlu G, Daim TU (2011) Using artificial neural network models in stock market index prediction. *Expert Syst Appl* 38(8):10389–10397
- Hinton G, Srivastava N, Swersky K (2012) Rmsprop: divide the gradient by a running average of its recent magnitude. *Neural Netw Mach Learn Coursera Lect* 6e:13
- Hochreiter S, Schmidhuber J (1997a) Long short-term memory. *Neural Comput* 9(8):1735–1780
- Hochreiter S, Schmidhuber J (1997b) LSTM can solve hard long time lag problems. In: Mozer MC, Jordan MI (eds) *Advances in neural information processing systems*. MIT Press, Cambridge, MA, pp 473–479
- Hu H, Ao Y, Bai Y, Cheng R, Xu T (2020) An improved Harris’s Hawks optimization for SAR target recognition and stock market index prediction. *IEEE Access* 8:65891–65910
- Huang Y, Gao Y, Gan Y, Ye M (2021) A new financial data forecasting model using genetic algorithm and long short-term memory network. *Neurocomputing* 425:207–218
- Hyndman RJ, Koehler AB (2006) Another look at measures of forecast accuracy. *Int J Forecast* 22(4):679–688
- Ibrahim D (2016) An overview of soft computing. *Procedia Comput Sci* 102:34–38
- Jamous R, ALRahhal H, El-Dariby M (2021) A new ANN-particle swarm optimization with center of gravity (ANN-PSOCOg) prediction model for the stock market under the effect of COVID-19. *Sci Program* 2021:1–17
- Ji Y, Liew AWC, Yang L (2021) A novel improved particle swarm optimization with long-short term memory hybrid model for stock indices forecast. *IEEE Access* 9:23660–23671
- Kamalov F (2020) Forecasting significant stock price changes using neural networks. *Neural Comput Appl* 32(23):17655–17667
- Kennedy J, Eberhart R (1995, November) Particle swarm optimization. In: Proceedings of ICNN’95-international conference on neural networks, vol 4. IEEE, pp 1942–1948
- Kim S, Kim H (2016) A new metric of absolute percentage error for intermittent demand forecasts. *Int J Forecast* 32(3):669–679
- Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. *arXiv preprint arXiv:1412.6980*
- Kumar G, Jain S, Singh UP (2021a) Stock market forecasting using computational intelligence: a survey. *Arch Comput Methods Eng* 28(3):1069–1101

- Kumar G, Jain S, Singh UP (2021b, March) Neuro-fuzzy and particle swarm optimization based hybrid approach for stock price forecasting. In: 2021b international conference on emerging smart computing and informatics (ESCI). IEEE, pp 753–758
- Kumar G, Singh UP, Jain S (2021c) Hybrid evolutionary intelligent system and hybrid time series econometric model for stock price forecasting. *Int J Intell Syst.* <https://doi.org/10.1002/int.22495>
- Kumar R, Kumar P, Kumar Y (2021d) Integrating big data driven sentiments polarity and ABC-optimized LSTM for time series forecasting. *Multimed Tools Appl.* <https://doi.org/10.1007/s11042-021-11029-1>
- Li AW, Bastos GS (2020) Stock market forecasting using deep learning and technical analysis: a systematic review. *IEEE Access* 8:185232–185242
- Lin CS, Chiu SH, Lin TY (2012) Empirical mode decomposition-based least squares support vector regression for foreign exchange rate forecasting. *Econ Model* 29(6):2583–2590
- Lin Y, Liu H, Xie G, Zhang Y (2018) Time series forecasting by evolving deep belief network with negative correlation search. In: 2018 Chinese automation congress (CAC). IEEE, pp 3839–3843
- Liu R, Liu L (2019) Predicting housing price in China based on long short-term memory incorporating modified genetic algorithm. *Soft Comput* 23(22):11829–11838
- Liu H, Long Z (2020) An improved deep learning model for predicting stock market price time series. *Digital Signal Process* 102:102741
- Liu F, Wang J (2012) Fluctuation prediction of stock market index by Legendre neural network with random time strength function. *Neurocomputing* 83:12–21
- Livieris IE, Pintelas E, Pintelas P (2020) A CNN–LSTM model for gold price time-series forecasting. *Neural Comput Appl* 32(23):17351–17360
- Long J, Chen Z, He W, Wu T, Ren J (2020) An integrated framework of deep learning and knowledge graph for prediction of stock price trend: an application in Chinese stock exchange market. *Appl Soft Comput* 91:106205
- Mahajan S, Mittal N, Pandit AK (2021) Image segmentation using multilevel thresholding based on type II fuzzy entropy and marine predators algorithm. *Multimedia Tools Appl* 80(13):19335–19359
- Mahajan S, Abualigah L, Pandit AK (2022a) Hybrid arithmetic optimization algorithm with hunger games search for global optimization. *Multimed Tools Appl* 81:28755–28778
- Mahajan S, Abualigah L, Pandit AK, Altalhi M (2022b) Hybrid Aquila optimizer with arithmetic optimization algorithm for global optimization tasks. *Soft Comput* 26(10):4863–4881
- Mahajan S, Abualigah L, Pandit AK, Nasar A, Rustom M, Alkhazaleh HA, Altalhi M (2022c). Fusion of modern meta-heuristic optimization methods using arithmetic optimization algorithm for global optimization tasks. *Soft Comput*, 1–15
- Mahajan S, Nayyar A, Raina A, Singh SJ, Vashishtha A, Pandit AK (2022d) A Gaussian process-based approach toward credit risk modeling using stationary activations. *Concurr Comput: Pract Exp* 34(5):e6692
- Menkhoff L (1997) Examining the use of technical currency analysis. *Int J Financ Econ* 2(4):307–318
- Nabipour M, Nayyeri P, Jabani H, Mosavi A, Salwana E (2020) Deep learning for stock market prediction. *Entropy* 22(8):840
- Nickabadi A, Ebadzadeh MM, Safabakhsh R (2011) A novel particle swarm optimization algorithm with adaptive inertia weight. *Appl Soft Comput* 11(4):3658–3670
- NSDL (National securities Depository limited). Retrieved from, <https://nsdl.co.in/about/statistics.php>. Accessed on: 15 Oct 2021
- Pal SS, Kar S (2022) Fuzzy transfer learning in time series forecasting for stock market prices. *Soft Comput* 26:6941–6952
- Pang X, Zhou Y, Wang P, Lin W, Chang V (2020) An innovative neural network approach for stock market prediction. *J Supercomput* 76(3):2098–2118
- Panigrahi BK, Pandi VR, Das S (2008) Adaptive particle swarm optimization approach for static and dynamic economic load dispatch. *Energy Convers Manage* 49(6):1407–1415
- Peng L, Liu S, Liu R, Wang L (2018) Effective long short-term memory with differential evolution algorithm for electricity price prediction. *Energy* 162:1301–1314
- Peng L, Zhu Q, Lv SX, Wang L (2020) Effective long short-term memory with fruit fly optimization algorithm for time series forecasting. *Soft Comput* 24:15059–15079. <https://doi.org/10.1007/s00500-020-04855-2>
- Rather AM, Agarwal A, Sastry VN (2015) Recurrent neural network and a hybrid model for prediction of stock returns. *Expert Syst Appl* 42(6):3234–3241
- Ren G, Cao Y, Wen S, Huang T, Zeng Z (2018) A modified Elman neural network with a new learning rate scheme. *Neurocomputing* 286:11–18
- Roodschild M, Sardinas JG, Will A (2020) A new approach for the vanishing gradient problem on sigmoid activation. *Prog Artif Intell* 9(4):351–360
- Ruder S (2016) An overview of gradient descent optimization algorithms. arXiv preprint [arXiv:1609.04747](https://arxiv.org/abs/1609.04747)
- Sahoo S, Mohanty MN (2020) Stock market price prediction employing artificial neural network optimized by gray wolf optimization. In: Patnaik S, Ip A, Tavana M, Jain V (eds) *New paradigm in decision science and management*. Springer, Singapore, pp 77–87
- Sakshi S, Kumar R (2019) a neuro-genetic technique for pruning and optimization of ANN weights. *Appl Artif Intell* 33(1):1–26
- Sands TM, Tayal D, Morris ME, Monteiro ST (2015, May) Robust stock value prediction using support vector machines with particle swarm optimization. In: 2015 IEEE congress on evolutionary computation (CEC). IEEE, pp 3327–3331
- Saxe AM, McClelland JL, Ganguli S (2013) Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. arXiv preprint [arXiv:1312.6120](https://arxiv.org/abs/1312.6120)
- Schaul T, Antonoglou I, Silver D (2013) Unit tests for stochastic optimization. arXiv preprint [arXiv:1312.6055](https://arxiv.org/abs/1312.6055)
- ShahvaroughiFarahani M, RazaviHajiagha SH (2021) Forecasting stock price using integrated artificial neural network and metaheuristic algorithms compared to time series models. *Soft Comput* 25(13):8483–8513
- Shi Y, Eberhart R (1998, May) A modified particle swarm optimizer. In: 1998 IEEE international conference on evolutionary computation proceedings. IEEE world congress on computational intelligence (Cat. No. 98TH8360). IEEE, pp 69–73
- Technical Indicators (2020) Technical analysis library in python. Retrieved from <https://technical-analysis-library-inpython.readthedocs.io/en/latest/ta.html>. Accessed 30 May 2020
- Theil H (1966) *Applied economic forecasting*. North-Holland Pub. Co., Amsterdam
- Ulke V, Sahin A, Subasi A (2018) A comparison of time series and machine learning models for inflation forecasting: empirical evidence from the USA. *Neural Comput Appl* 30(5):1519–1527
- Van Houdt G, Mosquera C, Napoles G (2020) A review on the long short-term memory model. *Artif Intell Rev* 53(8):5929–5955
- Wang P, Zhao J, Gao Y, Sotelo MA, Li Z (2020) Lane work-schedule of toll station based on queuing theory and PSO-LSTM model. *IEEE Access* 8:84434–84443
- Wang Y, Liu Y, Wang M, Liu R (2018, October) LSTM model optimization on stock price forecasting. In: 2018 17th international symposium on distributed computing and applications for business engineering and science (dcabes). IEEE, pp 173–177



- Xiong T, Bao Y, Hu Z, Chiong R (2015) Forecasting interval time series using a fully complex-valued RBF neural network with DPSO and PSO algorithms. *Inf Sci* 305:77–92
- Yan W (2012) Toward automatic time-series forecasting using neural networks. *IEEE Trans Neural Netw Learn Syst* 23(7):1028–1039
- Yazan E, Talu MF (2017, September) Comparison of the stochastic gradient descent based optimization techniques. In: 2017 international artificial intelligence and data processing symposium (IDAP). IEEE, pp 1–5
- Zhang P, Ci B (2020) Deep belief network for gold price forecasting. *Resour Policy* 69:101806

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.