Granular Computing (2023) 8:1645–1658
https://doi.org/10.1007/s41066-023-00389-8

ORIGINAL PAPER

A new training algorithm for long short-term memory artificial neural network based on particle swarm optimization

Turan Cansu[1] · Emine Kolemen[1] · Özlem Karahasan[1] · Eren Bas[1] · Erol Egrioglu[1]

Received: 4 May 2023 / Accepted: 27 May 2023 / Published online: 11 June 2023

Abstract

Long short-term memory deep artificial neural network is the most commonly used artificial neural network in the literature to solve the forecasting problem, and it is usually trained with the Adam algorithm, which is a derivative-based method. It is known that derivative-based methods are adversely affected by local optimum points and training results can have large variance due to their random initial weights. In this study, a new training algorithm is proposed, which is less affected by the local optimum problem and has a lower variance due to the random selection of initial weights. The proposed new training algorithm is based on particle swarm optimization, which is an artificial intelligence optimization method used to solve the numerical optimization problem. Since particle swarm optimization does not need the derivative of the objective function and it searches in the random search space with more than one solution point, the probability of getting stuck in the local optimum problem is lower than the derivative algorithms. When the proposed training algorithm is based on particle swarm optimization, the probability of getting stuck in the local optimum problem is very low. In the training algorithm, the restart strategy and the early stop condition are included so that the algorithm eliminates the overfitting problem. To test the proposed training algorithm, 10-time series obtained from FTSE stock exchange data sets are used. The proposed training algorithm is compared with Adam's algorithm and other ANNs using various statistics and statistical hypothesis tests. The application results show that the proposed training algorithm improves the results of long short-term memory, it is more successful than the Adam algorithm, and the long short-term memory trained with the proposed training algorithm gives superior forecasting performance compared to other ANN types.

Keywords Training algorithm · Long short-term memory · Forecasting · Particle swarm optimization

1 Introduction

The solution to the forecasting problem can be realized by many different methods. It is one of the most widely used scientific methods to obtain forecasts with a model created over the historical values of the time series using lagged variables. The models can be linear or non-linear. In particular, forecasting models created with artificial neural networks, where nonlinear designs are flexible and can change based on data thanks to hidden layers, have become popular in recent years. Although artificial neural networks are flexible approaches and data-driven methods, they have problematic aspects in solving the forecasting problem due to problems such as overfitting problems, local optimum traps and difficulties in determining hyperparameters. The most widely used multilayer perceptron has recently been replaced by deep artificial neural networks. Recurrent deep artificial neural networks are frequently preferred for forecasting purposes due to their structure that allows time dependence by the time series problem. Long short-term memory (LSTM), which offers a solution to the disappearing and exploding gradient problem encountered in recurrent deep networks by using gates in its architecture, is the most commonly used deep recurrent artificial neural network. While LSTM gets rid of the problems of derivative-based training algorithms thanks to gates, it invites overfitting problems by increasing the number of parameters. The literature of LSTM is given below and it is understood that it is used in solving forecasting problems that arise in different fields.

✉ Erol Egrioglu
erol.egrioglu@giresun.edu.tr

[1] Faculty of Arts and Science, Department of Statistics, Giresun University, 28200 Giresun, Turkey

The LSTM artificial neural network, which was first introduced by Hocreiter and Schmidhuber (1997), is widely used in forecasting studies. When the studies are examined, it has been determined that LSTM has started to be used intensively in fields such as energy, hydrology, economy and health. Fischer and Krauss (2018), Yao et al. (2018a; b), Seviye et al. (2018), Zhang and Yang (2019), Cen and Wang (2019), Vidya and Hari (2020), Yadav et al. (2020), Liu et al. (2021), Tang et al. (2022), Firouzjaee and Khaliliyan (2022), Kumar et al. (2022a, b) and Peng et al. (2020) used LSTM in their study. Livieris et al. (2020), Pirani et al. (2022), Ensafi et al. (2022), Freeboroug and Zyl (2022), Arunkumar et al. (2022), Kumar et al. (2022a, b), Sirisha et al. (2022) and Singha and Panse (2022) compared LSTM with different methods in their studies. In addition, LSTM, classical methods and hybrid approaches in which they are combined are used to solve the forecasting problem. A hybrid method was presented by Quadir et al. (2022). Many studies have been carried out in the field of engineering using LSTM. Some of these were proposed by Dong et al. (2017), Zhang et al. (2018), Weng et al. (2019), Huang et al (2022), Senanayake et al. (2022) and Silka et al. (2022). In addition, LSTM was compared with different forecasting methods in the literature by Barrera-Animas et al. (2022). When the studies are examined, it is seen that LSTM is used in the field of health as in many other fields. Chimmula and Zhang (2020) used LSTM to solve health data. In the field of tourism, there are many studies in which LSTM is used. The studies of Bi et al. (2020), Zhang et al. (2021) and Kumar et al. (2022a, b) can be given as an example of these studies. Solgi et al. (2021), Du et al. (2022a, b), Kilinc and Yurtsever (2022) and Li et al. (2022) used the LSTM method in the field of hydrology. Jiang and Hu (2018), Gong et al. (2022), Bilgili et al. (2022) and Karasu and Altan (2022) used LSTM in the field of energy. Besides, Zha et al. (2022), Yazici et al. (2022) and Ning et al. (2022) used the LSTM method in the field of energy. Tian et al. (2018), Liu et al. (2018), Karevan and Kim and Cho (2019), Karevan and Suykens (2020), Veeramsetty et al. (2021), Yu et al. (2022) and Du et al. (2022a, b) used hybrid approaches based on LSTM. Artificial intelligence optimization algorithms have been used in the training of LSTM. Genetic algorithm (GA) and particle swarm optimization (PSO) is the most preferred algorithms. Chen et al. (2018), Chung and Shin (2018) and Stajkowski et al. (2020) used GA as an optimization technique for the LSTM. Moalla et al. (2017), Yao et al. (2018a, b), Shao et al. (2019), Qiu et al. (2020) and Gundu and Simon (2021) used PSO for the training of LSTM. Bas et al. (2021) PSO used for training of Pi-Sigma. Liu and Song (2022) used granular neural networks. Fan et al. (2021), used a deep-learning approach for forecasting. Fuzzy inference systems and fuzzy time series are used for forecasting. Chen and Jian (2017), Chen and Jian (2017), Zeng et al. (2019), Chen et al. (2019), Bas et al. (2022a), Egrioglu et al. (2022), Pant and Kumar (2022a) and Pant and Kumar (2022b) used inference systems and fuzzy time series in their study.

The motivation of this study is the absence of a training algorithm in the literature for training an LSTM network designed for the solution of the forecasting problem, which reduces the variation due to the initial random weights based on PSO and is less affected in overfitting problems. The contributions of this study are given below:

- A training algorithm suitable for the designed LSTM is proposed for the forecasting problem.
- As the proposed training algorithm is based on PSO, it is less likely to get caught in local optimum traps.
- In the proposed training algorithm, different from other algorithms using PSO, the variability due to random starting weights has been reduced with a strategy called the restart strategy, and the results have been made more stable.
- In the proposed training algorithm, a solution to the overfitting problem is presented by defining the early stopping condition based on the proportional error, unlike other algorithms using PSO.
- Since the proposed algorithm does not need derivatives, it does not have the problem that the derivative cannot be calculated in certain regions of the search space.

In the second part of the study, the LSTM architecture, which is specific to the LSTM artificial neural network and forecasting problem, is given. In the third part of the study, the proposed new training algorithm is introduced. In the fourth chapter, application results for the FTSE time series are given. In the fifth chapter, the results and discussion are given.

2 LSTM artificial neural network for forecasting

LSTM is a deep artificial neural network with feedback connections formed by neurons containing gates. In the LSTM deep neural network, the first and most important feature to understand is the structure of an LSTM neuron. The structure of an LSTM cell is given in Fig. 1. There are input, forget, cell candidate and output gates in an LSTM neuron. Each gate in the LSTM cell has its input weights, recurrent weights and biases. Input gate, forget gate and cell candidate outputs are calculated as in the following equations.

$$i_t = \sigma(W_i x_t + R_i h_{t-1} + b_i) \tag{1}$$

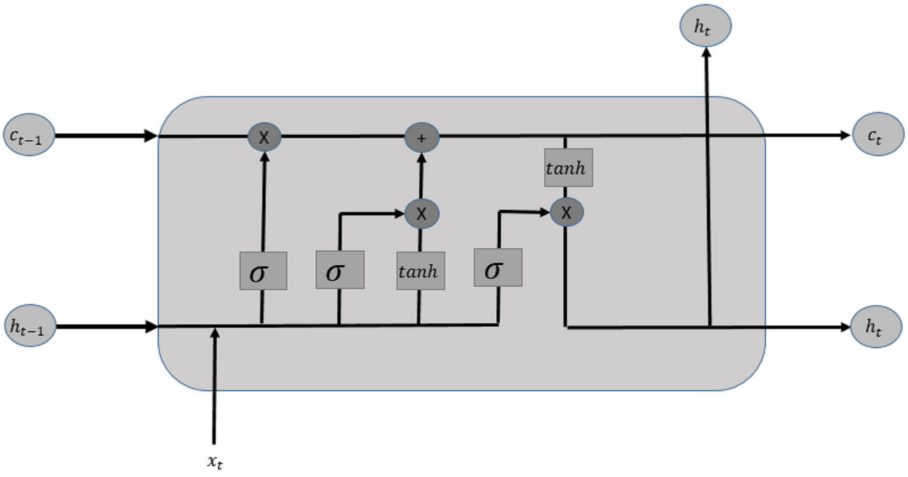


Fig. 1 The structure of the LSTM cell

$$f_t = \sigma(W_f x_t + R_f h_{t-1} + b_f) \quad (2)$$

$$g_t = \sigma(W_g x_t + R_g h_{t-1} + b_g) \quad (3)$$

After calculating the outputs of the input gate, forget gate and cell candidate gate by using Eqs. (1–3), the cell state value is calculated as in Eq. (4) using the previous cell state and the outputs of these gates.

$$c_t = f_t * c_{t-1} + i_t * g_t \quad (4)$$

The output of the output gate is calculated with the (5) formula. Then, the hidden state value is calculated by using Eq. (6) and the output gate output and the cell state outputs.

$$o_t = \sigma(W_o x_t + R_o h_{t-1} + b_o) \quad (5)$$

$$h_t = o_t * \tanh(c_t) \quad (6)$$

As a result, the inputs of an LSTM are inputs of the model (x_t), cell state and hidden state with one step delay (c_{t-1}, h_{t-1}) while the output of the LSTM cell is cell state and hidden state (c_t, h_t). Different LSTM deep neural networks can be revealed by different designs of LSTM cells. The architectural structure used in this study for the single variable time series forecasting problem is given in Fig. 2. In this architecture created for the forecasting problem, the inputs of the system are the lagged variables of the time series (y_t).

The input of an LSTM cell is lagged variable of $x_t = (y_t, y_{t-1,\ldots,} y_{t-p+1})$ according to the time step in Fig. 2. For example, the input for the LSTM cell in the lower right corner of the architecture is $x_{t-1} = (y_{t-1}, y_{t-2,\ldots,} y_{t-p})$ in Fig. 2. The output of the LSTM deep recurrent artificial neural network is one step forecast of time series $(\hat{y}_t)$. The output of the network is calculated with the following formula and σ represents the logistic activation function.

$$\hat{y}_t = \sigma(W_{FC} h_t + b_{FC}) \quad (7)$$

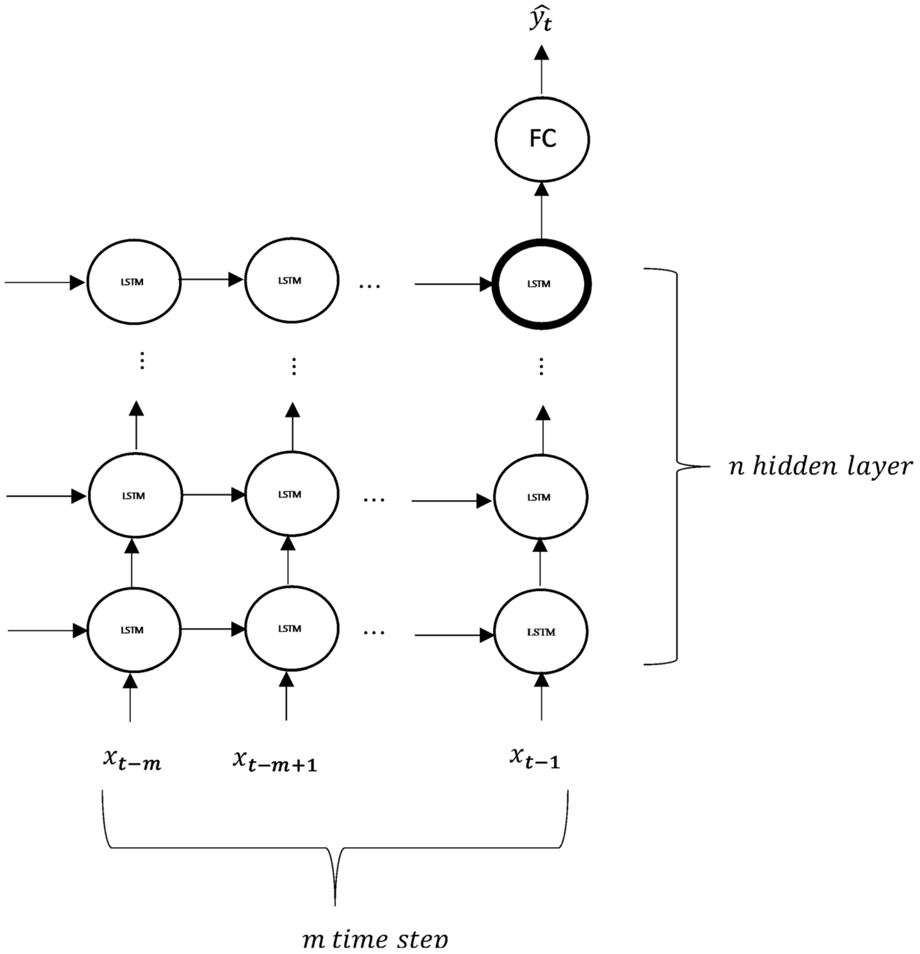
In Eq. (7), h_t is the output of the last thick-edged neuron in the architecture. The architecture in Fig. 2 has m time steps, n hidden layers and p inputs or features. The number of neurons in all hidden layers is equal to m. The weight and bias values of all LSTM cells in the same hidden layer are taken equally. This parameter sharing both reduces the number of parameters and enables a common LSTM cell that presents the same mathematical model in all time steps. These weights and bias values change in different hidden layers, that is, increasing the number of hidden layers increases the number of parameters of the network, while the number of time steps is not effective on the number of parameters.

Stochastic gradient descent algorithms can be used for training the LSTM given in Fig. 2. In stochastic gradient descent algorithms, the actual gradient is estimated by sampling. The real gradient is estimated by calculating the gradient over a random subgroup selected from the training data. One of the most commonly used stochastic gradient algorithms is Duchi et al. It is the AdaGrad algorithm proposed in (2011). In this algorithm, a scaling factor obtained from the outer product of the gradient vector is used. Another algorithm that can be used in the training of LSTM is the RMSROP algorithm. This algorithm is the minibatch version of the RROP algorithm that works using the sign of the gradient. The most frequently used training algorithm of LSTM, which is included in many packages, is the Adam algorithm given in Kingma and Ba (2014). The Adam algorithm is an algorithm with fast convergence properties. There are also more advanced versions of this algorithm in the literature. The Adam algorithm is implemented with the following steps.

Algorithm 1. Adam Algorithm

Step 1. Algorithms' initial parameter values are determined. The initial parameter values can be used as the default values of the Matlab package

Fig. 2 The architecture of LSTM deep recurrent artificial neural network for forecasting problem



programme Neural Netwrok toolbox as in other studies in the literature. α: Step sizeβ_1 and β_2: Exponential decay rates for the moment estimatesε: Correction term

Step 2. Initial values are set as zero.

$$m_0 = v_0 = t = 0$$

Step 3. Initial weight and bias values (θ_0) are determined. These values are realised by generating random numbers within a certain range.

Step 4. Do $t = t + 1$. The iteration counter is incremented by one.

Step 5. Calculate the following equations:

$$g_t = \nabla \mathrm{f} \quad (8)$$

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \quad (9)$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \quad (10)$$

$$\widehat{m}_t = m_t / (1 - \beta_1^t) \quad (11)$$

$$\widehat{v}_t = v_t / (1 - \beta_2^t) \quad (12)$$

$$\theta_t = \theta_{t-1} - \alpha \widehat{m}_t / (\sqrt{\widehat{v}_t} + \varepsilon) \quad (13)$$

In Eq. (10), g_t^2 is a vector containing diagonal elements of the Hessian matrix. The calculations are performed sequentially and in order.

Step 6. Repeat Steps 4–5 until convergence is achieved. Convergence is achieved by resetting the gradient vector to zero.

3 A new training algorithm for LSTM based on particle swarm optimization

The common point of AdaGrad, RMSROP and Adam algorithms is that they work with derivative information and operate with a single initial parameter vector set. This situation increases the possibility of the methods being caught in local optimum traps and the methods can be greatly influenced by the selection of initial parameters. In this study, a new training algorithm based on particle swarm optimization is designed, which has a lower risk of getting caught in local optimum traps and reduces the originating variance in determining the initial random weights. The Proposed algorithm works with two strategies called restarting and early stopping. The restarting strategy reduces the variance calculated for the change in the results caused by the random variation of the initial parameter values. The early stop condition can provide a solution to the overfitting problem by working with a proportional error. With these strategies, Bas et al. (2022), successful training results were obtained for the simple deep recurrent neural network. Moreover, since the proposed training algorithm is based on particle swarm optimization, it explores the search space with a set of these sets instead of a single initial parameter set, increasing the chance of avoiding the local optimum trap. The proposed training algorithm is presented below in steps.

Algorithm 2. A new training algorithm for LSTM

Step 1. The parameters of the PSO algorithm are selected according to the handled network.

$c_1^{initial}$: The starting value of the cognitive coefficient, the default value is 1.

c_1^{final} : The ending value of the cognitive coefficient, the default value is 2.

$c_2^{initial}$: The starting value of the social coefficient, the default value is 2.

c_2^{final} : The ending value of the social coefficient, the default value is 1.

$w^{initial}$: The starting value for inertia weight, the default value is 0.4.

w^{final} : The ending value for inertia weight, the default value is 0.9.

vmaps: The bound value for the velocities, the default value is 1.

$limit1$: The limit value for the re-starting strategy, the default value is 30.

$limit2$: The limit value for the early stopping rule, the default value is 20.

$maxitr$: The maximum number of iterations, the default value is 1000.

pn: The number of particles, the default value is 30.

Counters for restarting and early stopping strategies and iteration counter are reset ($rsc = 0,\ esc = 0, t = 0$). It is also possible to use alternative values for selecting the initial parameters.

Step 2. Initial position values and initial velocity values in particle swarm optimization are created.

The positions of a particle constitute all the weights and bias values of an LSTM network. The total number of weights and biases is $4m(p + m + 1) + m + 1$ in an LSTM cell because the dimensions of weights and biases are $W_i : p \times m$, $R_i : m \times m$, $b_i : 1xm$, $W_f : p \times m$, $R_f : m \times m$, $b_f : 1xm$, $W_g : p \times m$, $R_g : m \times m$, $b_g : 1xm$, $W_o : p \times m$, $R_o : m \times m$, $b_o : 1xm$, $W_{FC} : m \times 1$ and $b_{FC} : 1 \times 1$). For the LSTM network given in Fig. 2, the total weight and number of biases vary according to the number of hidden layers. This number is $4m(p + m + 1) + 4(n - 1)m(2m + 1) + m + 1$. For example, for $n = 2, m = 3, p = 4$, the total number of weights and biases is 184 in the LSTM network. The weights and biases are generated from $[0, 1]$ intervals. All velocities are generated from $[-vmaps,\ vmaps]$ interval. $P_{i,j}^{(t)}$ is the j^th^ position of the i^th^ particle at the t^th^ iteration. $V_{i,j}^{(t)}$ is the j^th^ velocity of i^th^ particle at the t^th^ iteration.

Step 3. As the fitness function to be used for the minimization of the PSO, the mean square error criterion given below is chosen. At this stage, it is possible to choose a different criterion, different choices will not affect the operation of the algorithm, but may affect the results. "vmaps" can also be a value that controls whether parameters are changed too much or too little.

$$MSE_j^t = \frac{1}{ntrain}\sum_{t=1}^{ntrain}(y_t - \hat{y}_t)^2, j = 1, 2, \ldots, pn \tag{14}$$

Step 4. The iteration counter is increased $t = t + 1$.

Step 5. Pbest and gbest are constituted. Pbest is a memory of all particles in the solution set, equal to their initial positions in the first step. Pbest is checked at each iteration, and the positions of the developing particles are memorized in Pbest. In a given iteration, the elements of Pbest represent the best individual memories of all particles so far. "gbest" is the memory of the swarm and is the best particle of Pbest. In this step, gbest and pbest vectors are generated only for the first iteration. In subsequent iterations these vectors are updated.

Step 6. The values of the cognitive, social coefficients and inertia weight parameters are calculated by the Eqs. (15–17).

$$w^{(t)} = (w^{initial} - w^{final})\frac{maxitr - t}{maxitr} + w^{final} \tag{15}$$

$$c_1^{(t)} = (c_1^{final} - c_1^{initial})\frac{t}{maxitr} + c_1^{initial} \tag{16}$$

$$c_2^{(t)} = (c_2^{initial} - c_2^{final})\frac{maxitr - t}{maxitr} + c_2^{final} \tag{17}$$

Equations (15–17) are used to linearly decrease or increase the coefficients. These equations allow the strengthening of global search by focusing on the strength of local search at the beginning and the final point on one point.

Step 7. The new velocities and positions are calculated by using the Eqs. (18–20). The r_1 and r_2 random numbers are generated from $[0,1]$ intervals. In addition, Eq. 3 ensures that the velocities are fixed within a certain range.

$$V_{i,j}^{(t)} = w^{(t)} V_{i,j}^{(t-1)} + c_1^{(t)} r_1 \left(Pbest_{i,j}^{(t)} - P_{i,j}^{(t)} \right) + c_2^{(t)} r_2 \left(gbest_j^{(t)} - P_{i,j}^{(t)} \right) \tag{18}$$

$$V_{i,j}^{(t)} = \min(vmaps, \max\left(-vmaps, V_{i,j}^{(t)}\right)) \tag{19}$$

$$P_{i,j}^{(t)} = P_{i,j}^{(t-1)} + V_{i,j}^{(t)} \tag{20}$$

Step 8. Using Eq. (14), the fitness functions of all particles are calculated. A different function could have been selected as fitness function.

Step 9. Pbest and gbest are updated. The fitness values calculated for the new positions of the particles are compared with the corresponding fitness value of Pbest, and if there is a new particle that lowers the fitness value according to its memory in Pbest, the relevant line of Pbest is replaced with new positions. Otherwise, no changes are made. After Pbest, the fitness value of the line with the best fitness value of the current Pbest is compared with the fitness value of "gbest". If an improvement is achieved, "gbest" is changed, otherwise, it is not changed.

Step 10. The restarting strategy counter is increased $(rsc = rsc + 1)$ and checked. If the $rsc > limit1$ then all positions and velocities are re-generated and the rsc is taken as zero but "Pbest" and "gbest" are saved in this step.

Step 11. The early stopping rule is checked. The esc counter is increased depending on the following condition.

$$esc = \begin{cases} esc + 1, & if \dfrac{MSEbest^{(t)} - MSEbest^{(t-1)}}{MSEbest^{(t)}} < 10^{-3} \\ 0, & otherwise \end{cases} \tag{21}$$

The early stopping rule is $esc > limit2$. If the rule is satisfied, the algorithm is stopped otherwise go to Step 4.

Step 12. The state of reaching the maximum number of iterations is checked. If $t > maxitr$, the algorithm is stopped and the best solution is taken as "gbest". Otherwise, it returns to Step 6.

A flowchart is given in Fig. 3 for a better understanding of the proposed new training algorithm.

Another important problem of LSTM given in Fig. 2 is hyperparameter selection. The properties of the analyzed data set may be important for the selection of hyperparameters. In general, it is preferred to select the hyperparameter values from a determined set by trial and error. In this method, determining the possible values of the hyperparameters is important in terms of determining the optimum calculation time. In this study, possible values of hyperparameters are determined by considering the observation frequency of the time series of interest. The following algorithm based on data partition is used for hyperparameter selection. The hyperparameters of the LSTM are considered as m, n and p.

Algorithm 3. Hyperparameter selection processes

Step 1. The possible values of the hyperparameters are determined. These values are lower and upper bound values for the hyperparameter values. The possible range of these values can be chosen according to the components contained in the time series. For example, in time series with seasonality, the possible values for the number of inputs of the model can be taken to include the period of the series.

$m \in [m_1, m_2]$

$n \in [n_1, n_2]$

$p \in [p_1, p_2]$

Step 2. Learning samples are divided into three parts training, validation and test set in a block structure. The reason for using the block structure is to ensure that the test set consists of more up-to-date data due to time dependence in the time series. While the LSTM is being trained with the training data, the hyperparameters are selected with the validation set. The test set is used to compare the performance of LSTM with other methods. The data fragmentation method is given in Fig 4. In Fig. 4 $ntrain, nval$ and $ntest$ are the number of observations for training, validation and test sets, respectively.

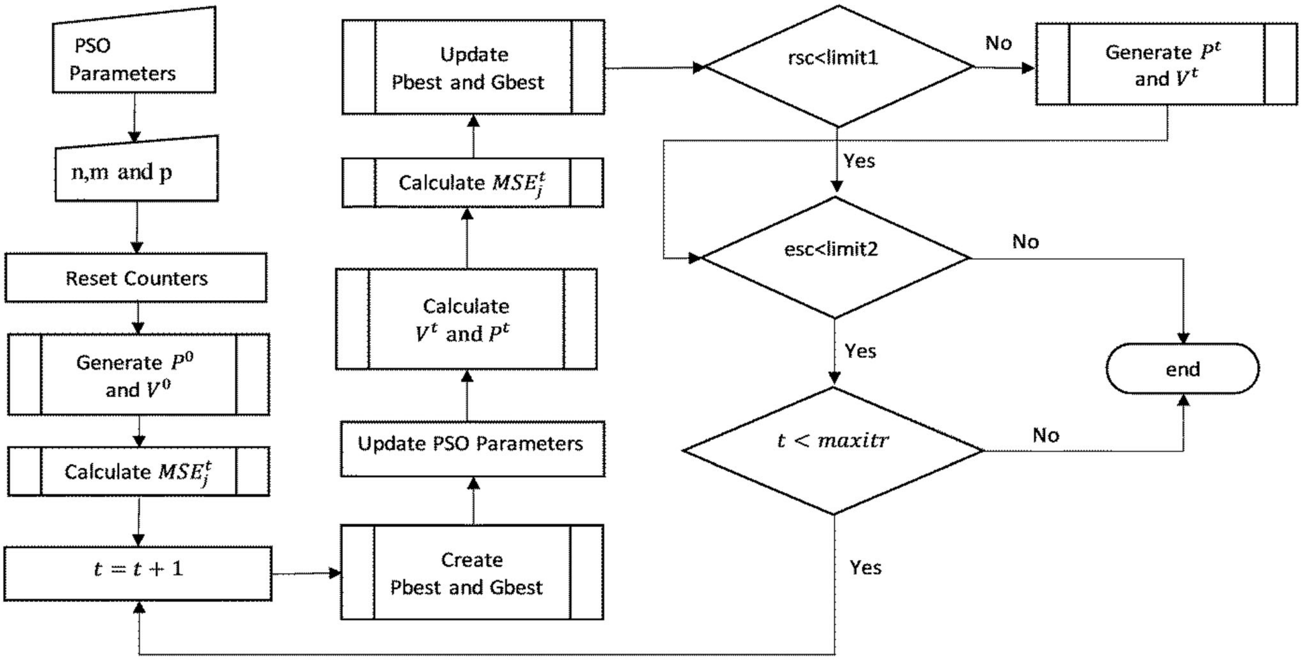


Fig. 3 Flowchart for the proposed training algorithm

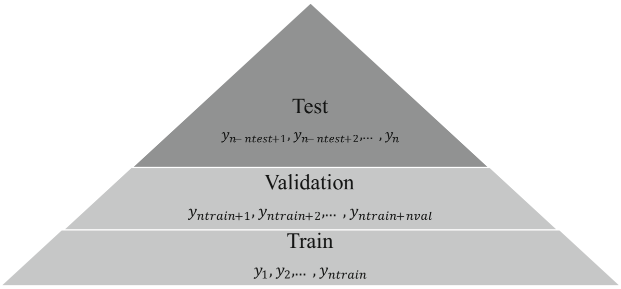

Test
$y_{n-ntest+1}, y_{n-ntest+2}, \cdots, y_n$
Validation
$y_{ntrain+1}, y_{ntrain+2}, \cdots, y_{ntrain+nval}$
Train
$y_1, y_2, \cdots, y_{ntrain}$

Fig. 4 The data partition for the hyperparameter selection

The lengths of the validity and test sets are determined by the frequency of observation of the time series. For example, for a five-day series, both the validation and test set can be taken as 20 to cover one month.

Step 3. Using Algorithm 2, train the LSTM using the training data for all possible hyperparameter values and calculate the root of mean square error values for the validation set.

Algorithm 2 is repeated $(m_2 - m_1 + 1) \times (n_2 - n_1 + 1) \times (p_2 - p_1 + 1)$ times in total according to possible parameter values.

$$RMSE_{val} = \sqrt{\frac{1}{nval} \sum_{k=ntrain+1}^{ntrain+nval} \left(y_t - \widehat{y}_t\right)^2} \tag{22}$$

The error measure in this step can be chosen as a different error measure instead of RMSE.

Step 4. The best hyperparameter values are selected. The best hyperparameter values $(mbest, nbest, pbest)$ are the hyperparameter values that give the lowest $RMSE_{val}$ value.

The flow diagram of the hyperparameter selection algorithm is given in Fig. 5.

4 Applications

The performance of the proposed new training algorithm in this study is investigated over ten-time series randomly selected from the closing prices of the FTSE 100 index. The FTSE 100 index is a market-capitalization-weighted index of UK-listed blue-chip companies. The index is part of the FTSE UK Series and is designed to measure the performance of the 100 largest companies traded on the London Stock Exchange. The information of the randomly selected time series is given in Table 1 and their graphs are given in Fig. 6. The random selection of the time series allows the closing values of different periods of the year to be included in the training, validation and test set each time and to make a more comprehensive and general comparison.

In the application, the performance of the proposed method is compared with the LSTM artificial neural

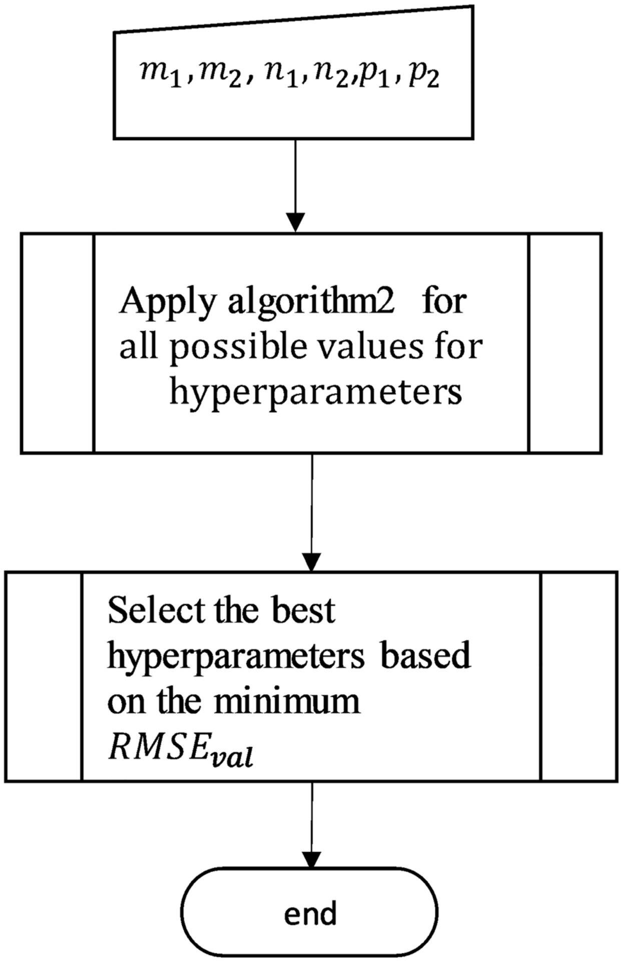

$m_1, m_2, n_1, n_2, p_1, p_2$
Apply algorithm2 for all possible values for hyperparameters
Select the best hyperparameters based on the minimum $RMSE_{val}$
end

Fig. 5 The flowchart of the hyperparameter selection algorithm

network trained with the Adam algorithm. When the most frequently used LSTM training algorithm in the literature is the Adam algorithm, the main success criterion of this study is to be able to reveal a more successful training algorithm than this algorithm. In addition, the performance of the proposed training algorithm is compared with Pi-Sigma, a high-order artificial neural network, and a simple recurrent deep artificial neural network (SRNN), which is a deep artificial neural network. Pi-Sigma and SRNN artificial neural networks are also trained with a training algorithm based on the PSO algorithm, similar to the proposed method.

In the implementation of all methods, the hyperparameter selection problem is performed with the hyperparameter selection algorithm used for the proposed method, ensuring a fair comparison. In the applications, the validation and test set lengths are taken as 60 for all series and all methods.

For the best hyperparameters, all methods are trained 30 times using different random initial weights, and the mean, standard deviation, and minimum and maximum statistics of the RMSE values obtained for the test set are calculated for these 30 replicates. In addition, the performance of the proposed method over these 30 replicates is compared with all methods separately using Wilcox's Signed Rank Test, and the p-values obtained for the significance obtained are given in the last column of the table. P-values less than the probability of type 1 error (α) determined in the last column indicate a statistically significant difference. All these results are presented in Table 2 for all series. In addition, the best hyperparameter values selected on the validation set for all methods are given in Table 3. Since Pi-Sigma ANN is not a deep artificial neural network, the number of hidden layers is fixed and 1.

When Table 2 is examined, it is seen that the most successful method for Series 1 is Pi-Sigma. For Series 1, it is seen that LSTM-ADAM produces lower mean RMSE values than the proposed method. It is also seen that LSTM-ADAM has a lower standard deviation and maximum statistics. When the proposed method is compared with LSTM-ADAM, it is understood that LSTM-ADAM has a statistically significant difference from the proposed

Table 1 The information about the randomly selected time series from FTSE 100 closing prices

Series	Number of observations	Training set dates	Validation set dates	Test set dates
Series 1	500	01.11.2019–05.05.2021	06.05.2021–27.07.2021	28.07.2021–20.10.2021
Series 2	500	22.05.2020–12.11.2021	15.11.2021–07.02.2022	08.02.2022–05.02.2022
Series 3	500	01.10.2015–27.03.2017	28.03.2017–21.06.2017	22.06.2017–15.09.2017
Series 4	500	09.06.2020–30.11.2021	01.12.2021–23.02.2022	24.02.2022–23.05.2022
Series 5	500	01.10.2018–02.04.2020	03.04.2020–29.06.2020	30.06.2020–22.09.2020
Series 6	500	31.07.2015–24.01.2017	25.01.2017–19.04.2017	20.04.2017–14.07.2017
Series 7	500	26.08.2016–20.02.2018	21.02.2018–17.05.2018	18.05.2018–10.08.2018
Series 8	500	28.03.2018–27.09.2019	30.09.2019–19.12.2022	20.12.2019–20.03.2020
Series 9	500	09.09.2020–04.03.2022	07.03.2022–30.05.2022	31.05.2022–24.08.2022
Series 10	500	25.09.2020–22.03.2022	23.03.2022–15.06.2022	16.06.2022–09.09.2022

Fig. 6 The randomly selected ten time series graphs from FTSE 100 Closing Prices

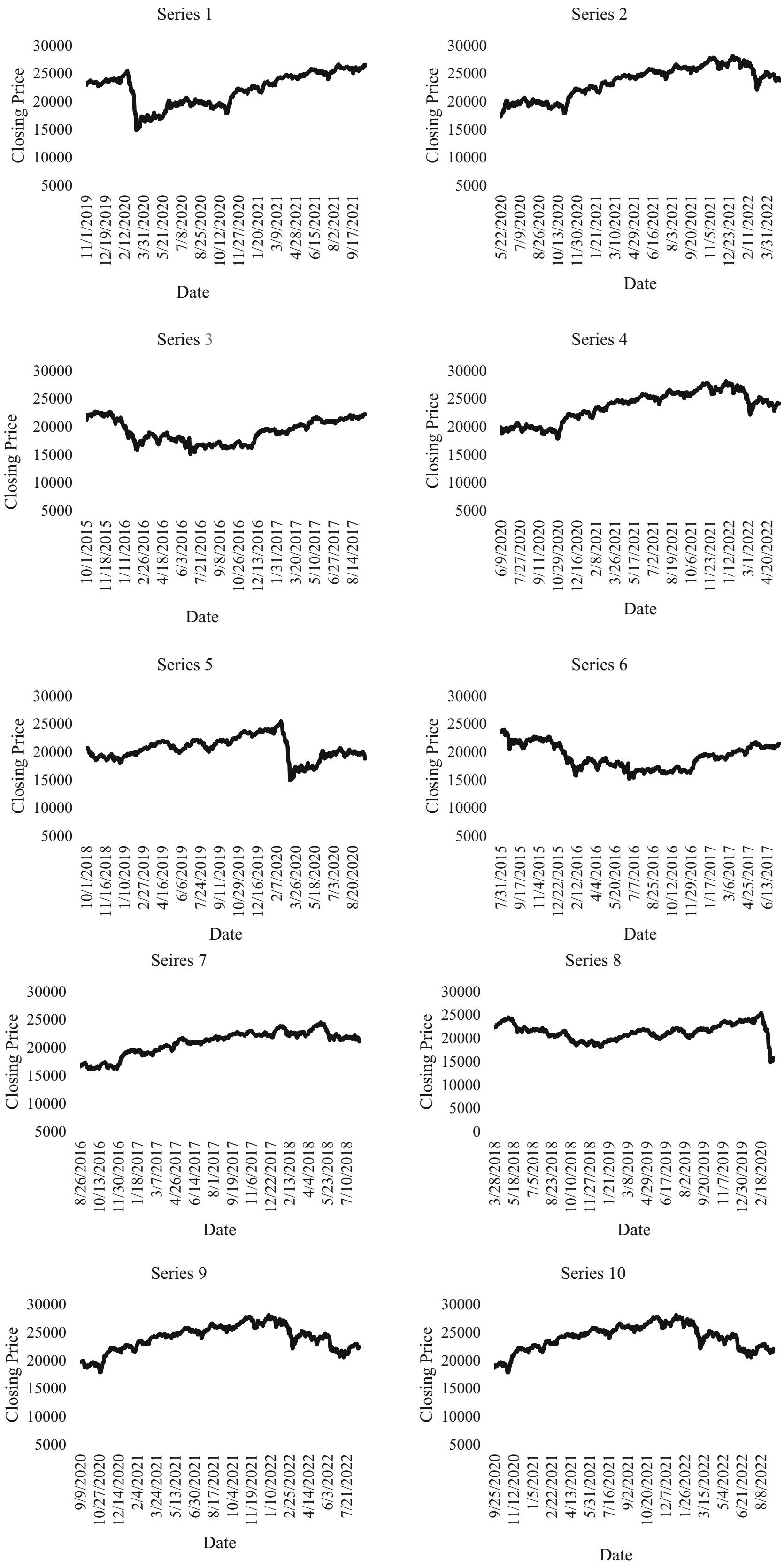


Table 2 The RMSE statistics for methods

Data Sets	Methods	Mean	Std. Dev	Minimum	Maximum	Wilcoxon test p-value
Series 1	LSTM-ADAM Hocreiter and Schmidhuber (1997)	0.008966	0.000005	0.008956	0.008973	0.000002
	Pi-Sigma Shin and Gosh (1991)	0.008857	0.000017	0.008827	0.008901	0.000002
	SRNN Bas et al. (2022b)	0.009131	0.000186	0.008896	0.009668	0.298944
	Proposed Method	0.009120	0.000095	0.008950	0.009368	
Series 2	LSTM-ADAM Hocreiter and Schmidhuber (1997)	0.021159	0.000116	0.020910	0.021366	0.000002
	Pi-Sigma Shin and Gosh (1991)	0.020419	0.000284	0.020089	0.021380	0.000002
	SRNN Bas et al. (2022b)	0.020426	0.000289	0.020032	0.021322	0.000005
	Proposed Method	0.020068	0.000035	0.020000	0.020150	
Series 3	LSTM-ADAM Hocreiter and Schmidhuber (1997)	0.007970	0.000039	0.007898	0.008052	0.025637
	Pi-Sigma Shin and Gosh (1991)	0.007856	0.000177	0.007671	0.008404	0.024308
	SRNN Bas et al. (2022b)	0.007828	0.000045	0.007747	0.007929	0.000005
	Proposed Method	0.007937	0.000089	0.007788	0.008153	
Series 4	LSTM-ADAM Hocreiter and Schmidhuber (1997)	0.022075	0.000296	0.021357	0.022650	0.000002
	Pi-Sigma Shin and Gosh (1991)	0.019593	0.000210	0.019122	0.020083	0.000160
	SRNN Bas et al. (2022b)	0.019532	0.000123	0.019373	0.019805	0.000148
	Proposed Method	0.019422	0.000061	0.019361	0.019649	
Series 5	LSTM-ADAM Hocreiter and Schmidhuber (1997)	0.013290	0.000016	0.013244	0.013317	0.000002
	Pi-Sigma Shin and Gosh (1991)	0.013166	0.000007	0.013146	0.013178	0.002105
	SRNN Bas et al. (2022b)	0.013167	0.000090	0.012941	0.013328	0.557743
	Proposed Method	0.013153	0.000033	0.013085	0.013286	
Series 6	LSTM-ADAM Hocreiter and Schmidhuber (1997)	0.010800	0.000005	0.010791	0.010816	0.000006
	Pi-Sigma Shin and Gosh (1991)	0.010690	0.000042	0.010578	0.010829	0.688359
	SRNN Bas et al. (2022b)	0.010660	0.000055	0.010558	0.010866	0.228880
	Proposed Method	0.010684	0.000089	0.010538	0.010843	
Series 7	LSTM-ADAM Hocreiter and Schmidhuber (1997)	0.012468	0.000104	0.012296	0.012642	0.000010
	Pi-Sigma Shin and Gosh (1991)	0.012228	0.000102	0.012057	0.012567	0.003854
	SRNN Bas et al. (2022b)	0.012327	0.000042	0.012219	0.012439	0.040702
	Proposed Method	0.012295	0.000072	0.011998	0.012394	
Series 8	LSTM-ADAM Hocreiter and Schmidhuber (1997)	0.034638	0.000038	0.034496	0.034654	0.000008
	Pi-Sigma Shin and Gosh (1991)	0.034349	0.000008	0.034342	0.034378	0.000002
	SRNN Bas et al. (2022b)	0.034580	0.000019	0.034516	0.034601	0.012453
	Proposed Method	0.034563	0.000021	0.034523	0.034617	
Series 9	LSTM-ADAM Hocreiter and Schmidhuber (1997)	0.016315	0.000010	0.016305	0.016342	0.000097
	Pi-Sigma Shin and Gosh (1991)	0.016339	0.000057	0.016209	0.016412	0.958990
	SRNN Bas et al. (2022b)	0.016331	0.000015	0.016307	0.016386	0.404835
	Proposed Method	0.016341	0.000028	0.016303	0.016397	
Series 10	LSTM-ADAM Hocreiter and Schmidhuber (1997)	0.014149	0.000064	0.014002	0.014252	0.000771
	Pi-Sigma Shin and Gosh (1991)	0.014354	0.000243	0.014021	0.015187	0.000007
	SRNN Bas et al. (2022b)	0.014116	0.000000	0.014116	0.014117	0.001114
	Proposed Method	0.014100	0.000027	0.014029	0.014155	

method according to the Wilcoxon signed rank test and produces better forecasting results. Besides, it is seen that the proposed method for Series 1 and the results of SRNN have statistically equivalent performance according to Wilcoxon signed rank test.

When the results given for Series 2 in Table 2 are examined, it is understood that the proposed method has more successful RMSE statistics than all methods and these differences are statistically significant according to Wilcox's signed rank test. When the results for Series 3 are examined in Table 2, it is seen that the most successful

Table 3 The best hyperparameter values for all methods

Data sets	Methods	*pbest*	*nbest*	*hbest*
Series 1	LSTM-ADAM Hocreiter and Schmidhuber (1997)	5	1	5
	Pi-Sigma Shin and Gosh (1991)	1	1	4
	SRNN Bas et al. (2022b)	3	1	3
	Proposed Method	4	1	3
Series 2	LSTM-ADAM Hocreiter and Schmidhuber (1997)	5	5	5
	Pi-Sigma Shin and Gosh (1991)	3	1	5
	SRNN Bas et al. (2022b)	4	1	3
	Proposed Method	4	3	2
Series 3	LSTM-ADAM Hocreiter and Schmidhuber (1997)	5	5	5
	Pi-Sigma Shin and Gosh (1991)	4	1	5
	SRNN Bas et al. (2022b)	3	1	1
	Proposed Method	4	1	2
Series 4	LSTM-ADAM Hocreiter and Schmidhuber (1997)	1	5	1
	Pi-Sigma Shin and Gosh (1991)	3	1	2
	SRNN Bas et al. (2022b)	4	1	3
	Proposed Method	2	2	1
Series 5	LSTM-ADAM Hocreiter and Schmidhuber (1997)	4	2	4
	Pi-Sigma Shin and Gosh (1991)	1	1	4
	SRNN Bas et al. (2022b)	5	1	2
	Proposed Method	5	2	2
Series 6	LSTM-ADAM Hocreiter and Schmidhuber (1997)	5	1	5
	Pi-Sigma Shin and Gosh (1991)	3	1	3
	SRNN Bas et al. (2022b)	3	1	1
	Proposed Method	3	1	2
Series 7	LSTM-ADAM Hocreiter and Schmidhuber (1997)	4	5	4
	Pi-Sigma Shin and Gosh (1991)	5	1	2
	SRNN Bas et al. (2022b)	3	1	1
	Proposed Method	3	1	3
Series 8	LSTM-ADAM Hocreiter and Schmidhuber (1997)	1	1	1
	Pi-Sigma Shin and Gosh (1991)	1	1	4
	SRNN Bas et al. (2022b)	1	3	2
	Proposed Method	3	5	1
Series 9	LSTM-ADAM Hocreiter and Schmidhuber (1997)	2	1	2
	Pi-Sigma Shin and Gosh (1991)	1	1	4
	SRNN Bas et al. (2022b)	5	5	2
	Proposed Method	2	2	3
Series 10	LSTM-ADAM Hocreiter and Schmidhuber (1997)	3	4	3
	Pi-Sigma Shin and Gosh (1991)	5	1	3
	SRNN Bas et al. (2022b)	3	5	2
	Proposed Method	2	1	3

method is SRNN and it produces statistically more successful forecasting results than the proposed method. In addition, it is seen that the proposed method produces more successful forecasting results than the LSTM-ADAM method and this difference is statistically significant according to the Wilcoxon signed-rank test. When the results for Series 4 are examined in Table 2, it is seen that the most successful method is the recommended method according to the mean, standard deviation and maximum statistics, only the pi-sigma method produces a lower RMSE than all other methods according to the minimum statistics. However, it is understood that the results produced by the proposed method have a statistically lower median than all other methods according to the Wilcoxon signed-rank test.

When the results for Series 5 are examined in Table 2, it is seen that the most successful method is the recommended method according to the mean, minimum and maximum statistics, only the Pi-sigma method is lower than all other methods according to the standard deviation statistics. However, it is understood that the results produced by the proposed method have a statistically lower median than all other methods except SRNN according to the Wilcoxon signed-rank test. It is understood that the forecasting performance of the proposed method and the SRNN method are equivalent according to the Wilcoxon signed-rank test.

When the results for Series 6 are examined in Table 2, it is seen that the most successful method is SRNN according to the mean, minimum and maximum statistics. However, it is understood that the forecasting performance produced by the proposed method is statistically equivalent to SRNN, Pi-sigma. In addition, it is seen that the proposed method produces more successful and statistically significant results than the LSTM-ADAM method.

When the results for Series 7 are examined in Table 2, it is seen that the most successful method is the recommended method according to the standard deviation, minimum and maximum statistics. Pi-sigma ANN produced lower mean statistics. In addition, Pi-sigma ANN produces forecasting results with a lower median according to Wilcox's signed rank test. In addition, it is seen that the proposed method produces more successful and statistically significant results in all statistics than LSTM-ADAM and SRNN methods according to the Wilcoxon signed-rank test. When the results for Series 8 are examined in Table 2, a situation similar to the results obtained for Series 7.

When the results for Series 9 are examined in Table 2, it is seen that the best results are obtained from the LSTM-ADAM algorithm. Besides, it is understood that the performance of the proposed method is equivalent to Pi-sigma and SRNN according to Wilcox's signed rank test. Finally, when the results for Series 10 in Table 2 are examined, it is seen that the best mean and maximum statistics results are obtained from the proposed method. In addition, it is understood that the SRNN method produces the best results according to the minimum and standard deviation statistics. It is understood that the method with the lowest median according to Wilcox's signed rank test is the recommended method and produces RMSE statistics with a statistically significant difference and a lower median. When Table 3 is examined, it is understood that the best hyperparameter values vary considerably according to the series and the applied method. Although the proposed method and LSTM-ADAM methods have the same artificial neural network structure, different training algorithms may cause different hyperparameter selections.

5 Conclusion and discussions

In this study, an LSTM architecture for solving the forecasting problem with LSTM and a new PSO-based training algorithm for solving LSTM in this architectural structure is proposed. The proposed training algorithm has superior features compared to the Adam algorithm, which is the most frequently used and derivative-based training algorithm in the literature. Since the proposed training algorithm is based on PSO and does not require derivatives, there is no problem in the search space regions where the error function, which is possible for the Adam algorithm, cannot be differentiated.

In addition, the restart strategy in the proposed method reduces the variation caused by the differences in the random selection of the starting weights and ensures the homogeneity of the results. It was expected and observed that the early stopping condition could cure the overfitting problems. All these findings are supported by the results obtained in the application. While the success of the proposed method in practice is 50%, Pi-Sigma 30%, SRNN 10% and LSTM-ADAM 10%. In addition, when the proposed method is compared with LSTM-ADAM, it is seen that the proposed method produces more successful forecasting results in the LSTM-ADAM method in 80% of all series. All these findings are supported by the results obtained using Wilcox's signed rank test. As a result, an effective and new training algorithm has been introduced for the solution of the univariate time series forecasting problem.

In future studies, it is planned to improve the results further by making adjustments in the LSTM architecture that will positively affect the forecasting performance. In addition, the use of different artificial intelligence optimization algorithms in the training algorithm is included in the plans of our future studies as another subject that should be investigated.

Author contributions TC: Methodology, Software. EK: Writing, Editing. OK: Methodology, Software. EB: Methodology, Conceptualization, Writing, Editing. EE: Methodology, Software, Writing, Conceptualization.

Funding The authors did not receive support from any organization for the submitted work.

Data availability All data analysed during this study are included in this published article's supplementary files.

Declarations

Conflict of interest The author declares that there is no conflict of interest toward the publication of this manuscript. The authors declare they have no financial interests.

Supplementary Information

The online version contains supplementary material available at https://doi.org/10.1007/s41066-023-00389-8.

References

Arunkumar KE, Kalaga DV, Kumar CMS, Kavaji M, Brenza TM (2022) Comparative analysis of Gated Recurrent Units (GRU), long Short-Term memory (LSTM) cells, autoregressive Integrated moving average (ARIMA), seasonal autoregressive Integrated moving average (SARIMA) for forecasting COVID-19 trends. Alex Eng J 61(10):7585–7603

Barrera Animas AY, Oyedele LO, Bilal M, Akinosho TD, Delgado JMD, Akanbi LA (2022) Rainfall prediction: a comparative analysis of modern machine learning algorithms for time-series forecasting. Mach Learn Appl 7:100204

Bas E, Yolcu U, Egrioglu E (2021) Intuitionistic fuzzy time series functions approach for time series forecasting. Granul Comput 6:619–629

Bas E, Egrioglu E, Karahasan O (2022a) A Pi-Sigma artificial neural network based on sine cosine optimization algorithm. Granul Comput 7:813–820

Bas E, Egrioglu E, Kolemen E (2022b) Training simple recurrent deep artificial neural network for forecasting using particle swarm optimization. Granul Comput 7(2):411–420

Bi JW, Liu Y, Li H (2020) Daily tourism volume forecasting for tourist attractions. Ann Tour Res 83:102923

Bilgili M, Arslan N, Şekertekin A, Yaşar A (2022) Application of long short-term memory (LSTM) neural network based on deep learning for electricity energy consumption forecasting. Turk J Electr Eng Comput Sci 30(1):140–157

Cen Z, Wang H (2019) Crude oil price prediction model with long short-term memory deep learning based on prior knowledge data transfer. Energy 169:160–171

Chen SM, Jian WS (2017) Fuzzy forecasting based on two-factors second-order fuzzy-trend logical relationship groups, similarity measures and PSO techniques. Inf Sci 391:65–79

Chen J, Xing H, Yang H, Xu L (2018) Network traffic prediction based on LSTM networks with genetic algorithm. In: Proceedings of International Conference on Signal and Information Processing, Networking and Computers, pp 411–419

Chen SM, Zou XY, Gunawan GC (2019) Fuzzy time series forecasting based on proportions of intervals and particle swarm optimization techniques. Inf Sci 500:127–139

Chimmula, Zhang (2020) Time series forecasting of COVID-19 transmission in Canada using LSTM networks. Chaos Solit. Fractals 135:109864

Chung H, Shin KS (2018) Genetic algorithm-optimized long short-term memory network for stock market prediction. Sustainability 10(10):3765

Dong D, Li XY, Sun FQ (2017) Life prediction of jet engines based on LSTM-recurrent neural networks. Prognostics And System Health Management Conference, pp 1–6

Du B, Huang S, Guo J, Tang H, Wang L, Zhou S (2022a) Interval forecasting for urban water demand using PSO optimized KDE distribution and LSTM neural networks. Appl Soft Comput 122:108875

Du J, Zheng J, Liang Y, Lu X, Klemes JJ, Varbanov PS, Shahzad K, Rashid MI, Ali AM, Liao Q, Wang B (2022b) A hybrid deep learning framework for predicting daily natural gas consumption. Energy 257:124689

Egrioglu E, Fildes R, Bas E (2022) Recurrent fuzzy time series functions approaches for forecasting. Granul Comput 7:163–170

Ensafi Y, Amin SH, Zhang G, Shah B (2022) Time-series forecasting of seasonal items sales using machine learning–A comparative analysis. Int J Inf Manag Data Insights 2(1):100058

Fan MH, Chen MY, Liao EC (2021) A deep learning approach for financial market prediction: utilization of Google trends and keywords. Granul Comput 6:207–216

Firouzjaee and Khaliliyan (2022) LSTM architecture for oil stocks prices prediction. arXiv

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

Freeborough, Zyl (2022) Investigating explainability methods in recurrent neural network architectures for financial time series data. Appl Sci 12(3):1427

Gong Y, Zhang X, Gao D, Li H, Yan L, Peng J, Huang Z (2022) State-of-health estimation of lithium-ion batteries based on improved long short-term memory algorithm. J Energy Storage 53:105046

Gundu V, Simon SP (2021) PSO–LSTM for short-term forecast of heterogeneous time series electricity price signals. J Ambient Intell Humaniz Comput 12:2375–2385

Hocreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780

Huang R, Wei C, Wang B, Yang J, Xu X, Wu S, Huang S (2022) Well performance prediction based on long short-term memory (LSTM) neural network. J Pet Sci Eng 208:109686

Jiang L, Hu G (2018) Day-ahead price forecasting for electricity market using long-short term memory recurrent neural network. In: Proceedings of IEEE International Conference on Control, Automation, Robotics and Vision, pp 949–954

Karasu S, Altan A (2022) Crude oil time series prediction model based on LSTM network with chaotic Henry gas solubility optimization. Energy 242:122964

Karevan, Suykens (2020) Transductive LSTM for time-series prediction: an application to weather forecasting. Neural Netw 125:1–9

Kilinc HC, Yurtsever A (2022) Short-term streamflow forecasting using hybrid deep learning model based on grey wolf algorithm for hydrological time series. Sustainability 14(6):3352

Kim TY, Cho SB (2019) Predicting residential energy consumption using CNN-LSTM neural networks. Energy 182:72–81

Kingma D, Ba JL (2014) Adam: a method for stochastic optimization. Comput Sci 1–15. arXiv: 1412.6980

Kumar G, Singh UP, Jain S (2022a) An adaptive particle swarm optimization-based hybrid long short-term memory model for stock price time series forecasting. Soft Comput 26(22):12115–12135

Kumar S, Chitradevi D, Rajan S (2022b) Stock price prediction using deep learning LSTM (long short-term memory). In: Proceedings of International Conference on Advance Computing and Innovative Technologies in Engineering, pp 1787–1791

Li Q, Yang Y, Yang L, Wang Y (2022) Comparative analysis of water quality prediction performance based on LSTM in the Haihe River Basin. China Environ Sci Pollut Res 30(3):7498–7509

Liu Y, Song M (2022) Few samples learning based on granular neural networks. Granul Comput 7:572–589

Liu H, Mi XW, Li YF (2018) Wind speed forecasting method based on deep learning strategy using empirical wavelet transform, long short-term memory neural network and Elman neural network. Energy Convers Manag 156:498–514

Liu K, Zhou J, Dong (2021) Improving stock price prediction using the long short-term memory model combined with online social networks. J Behav Exp Finance 30:100507

Livieris LE, Pintelas E, Pintelas P (2020) A CNN–LSTM model for gold price time-series forecasting. Neural Comput Appl 32(23):17351–17360

Moalla H, Elloumi W, Alimi AM (2017) H-PSO-LSTM: Hybrid LSTM trained by PSO for online handwriting identification. In: Proceedings of International Conference on Neural Information Processing, pp 41–50

Ning Y, Kazemi H, Tahmasebi P (2022) A comparative machine learning study for time series oil production forecasting: ARIMA, LSTM, and Prophet. Comput Geosci 164:105126

Pant M, Kumar S (2022a) Fuzzy time series forecasting based on hesitant fuzzy sets, particle swarm optimization and support vector machine-based hybrid method. Granul Comput 7:861–879

Pant M, Kumar S (2022b) Particle swarm optimization and intuitionistic fuzzy set-based novel method for fuzzy time series forecasting. Granul Comput 7(2):285–303

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

Pirani M, Thakkar P, Jivrani P, Bohara MH, Garg D (2022) A comparative analysis of arima, gru, lstm and bilstm on financial time series forecasting. In Proceedings of IEEE International Conference on Distributed Computing and Electrical Circuits and Electronics, pp 1–6

Qiu YY, Zhang Q. Lei M (2020) Forecasting the railway freight volume in China based on combined PSO-LSTM model. In: Proceedings of Journal of Physics: Conference Series, pp 012029

Quadir MdA, Kapoor S, Chris Junni AV, Sivaraman AK, Tee KF, Sabireen H, Janakiraman N (2022) Novel optimization approach for stock price forecasting using multi-layered sequential LSTM. Appl Soft Comput 134:109830

Senanayake S, Pradhan B, Alamri A, Park HJ (2022) A new application of deep neural network (LSTM) and RUSLE models in soil erosion prediction. Sci Total Environ 845:157220

Seviye L, Kong W, Qi J, Zhang J (2018) An improved long short-term memory neural network for stock forecast. In MATEC web of conferences, pp 01024

Shao B, Li M, Zhao Y, Bian G (2019) Nickel price forecast based on the LSTM neural network optimized by the improved PSO algorithm. Math Probl Eng 2019:1934796

Shin Y, Gosh J (1991) The Pi-Sigma network: an efficient higher order neural network for pattern classification and function approximation. In: Proceedings of the International Joint Conference on Neural Networks, Seattle, pp 13–18

Silka J, Wieczorek M, Wozniak M (2022) Recurrent neural network model for high-speed train vibration prediction from time series. Neural Comput Appl 34:1–14

Singha D, Panse C (2022) Application of different machine learning models for supply chain demand forecasting: comparative analysis. In: Proceedings of International Conference on Innovative Practices in Technology and Management, pp 312–318

Sirisha UM, Belavagi MC, Attigeri G (2022) Profit prediction using ARIMA, SARIMA and LSTM Models in time series forecasting, a comparison. IEEE Access 10:124715–124727

Solgi R, Loaciga HA, Kram M (2021) Long short-term memory neural network (LSTM-NN) for aquifer level time series forecasting using in-situ piezometric observations. J Hydrol 601:126800

Stajkowski S, Kumar D, Samui P, Bonakdari H, Gharabaghi B (2020) Genetic-algorithm-optimized sequential model for water temperature prediction. Sustainability 12(13):5374

Tang Y, Song Z, Zhu Y, Yuan H, Hou M, Ji J, Tang C, Li J (2022) A survey on machine learning models for financial time series forecasting. Neurocomputing 512:363–380

Tian C, Ma J, Zhang C, Zhan P (2018) A deep neural network model for short-term load forecast based on long short-term memory network and convolutional neural network. Energies 11(12):3493

Veeramsetty V, Chandra DR, Salkuti SR (2021) Short-term electric power load forecasting using factor analysis and long short-term memory for smart cities. Int J Circuit Theory Appl 49(6):1678–1703

Vidya GS, Hari VS (2020) Gold price prediction and modelling using deep learning techniques. In: Proceedings of IEEE Recent Advances in Intelligent Computational Systems, pp 28–31

Weng C, Liu S, Yang X, Peng L, Li X, Hu Y, Chi T (2019) A novel spatiotemporal convolutional long short-term neural network for air pollution prediction. Sci Total Environ 654:1091–1099

Yadav A, Ja CK, Sharan A (2020) Optimizing LSTM for time series prediction in Indian stock market. Procedia Comput Sci 167:2091–2100

Yao S, Lua L, Peng H (2018a) High-frequency stock trend forecast using LSTM Model. In: Proceedings of IEEE International Conference on Computer Science Education, pp 1–4

Yao Y, Han L, Wang J (2018b) LSTM-PSO: Long short-term memory ship motion prediction based on particle swarm optimization. In: Proceedings of the IEEE CSAA Guidance, Navigation and Control Conference (CGNCC), pp 1–5

Yazici I, Beyca OF, Delen D (2022) Deep-learning-based short-term electricity load forecasting: a real case application. Eng Appl Artif Intell 109:104645

Yu L, Liang S, Chen R, Lai KK (2022) Predicting monthly biofuel production using a hybrid ensemble forecasting methodology. Int J Forecast 38(1):3–20

Zeng S, Chen SM, Teng MO (2019) Fuzzy forecasting based on linear combinations of independent variables, subtractive clustering algorithm and artificial bee colony algorithm. Inf Sci 484:350–366

Zha W, Liu Y, Wan Y, Luo R, Li D, Yang S, Xu Y (2022) Forecasting monthly gas field production based on the CNN-LSTM model. Energy 260:124889

Zhang Y, Yang S (2019) Prediction on the highest price of the stock based on PSO-LSTM Neural Network. In: Proceedings of International Conference on Electronic Information Technology and Computer Engineering, pp 1565–1569

Zhang J, Zhu Y, Zhang X, Ye M, Yang J (2018) Developing a long short-term memory (LSTM) based model for predicting water table depth in agricultural areas. J Hydrol 561:918–929

Zhang H, Xu J, Qian L, Qiu J (2021) Prediction of the COVID-19 spread in China based on long short-term memory network. J Phys 2138:012015

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