Journal of Computer Science

Original Research Paper

Forecasting of Multistep Multivariate Financial Data Through GSO Algorithm Infused Vanilla LSTM Model

[1]Nikhitha Pai, [1]Ilango Velchamy and [2]Nithya B Ramesh

[1]Department of MCA, CMR Institute of Technology, Bengaluru, India
[2]Department of MCA, New Horizon College of Engineering, Bengaluru, India

Article history
Received: 24-03-2023
Revised: 26-04-2023
Accepted: 02-05-2023

Corresponding Author:
Nithya B Ramesh
Department of MCA, New Horizon College of Engineering, Bengaluru, India
Email: nithya.boopalan@gmail.com

Abstract: Predicting time series in the financial domain requires using models that can do sequence processing tasks. One such model is the LSTM model which belongs to the family of deep learning networks. The performance measure of LSTM in a regression task is to bring down the error in the predicted values. In this study, a novel fused LSTM model integrating a glowworm optimization algorithm helps to do a multistep prediction of close stock prices of select companies from the FMCG sector of BSE and the consumables sector of NSE 50. The fused GSO algorithms are applied to LSTMs to bring down the difference between the predicted and real values. The reason for using GSO Infused LSTM implementation in this study is that pure LSTMs cannot give very good prediction results. Once the topology of the LSTM is tuned well the accuracy of prediction results improves considerably. This has been achieved with the GSO algorithm infused in the LSTM model. The justification for using optimization through GSO for the optimization of LSTMs is that it takes lesser time to converge than the gradient descent and gives faster results of better solutions compared to commonly used approaches like Grid search and Random search. Glowworm swarm optimization can also be used for easy convergence to optima in the case of multimodal functions. The results of the experiment show that there is a considerable reduction in the difference between actual and predicted values in the case of LSTMs interfaced with GSO when compared with other regressors like support vector regressor, KNN, random forest, and stacked LSTM and GRU models. For the Training set of data, the mean RMSE values for the different models were obtained as 270.17 for SVR, 21.16 for the random forest, 90.72 for KNN, 44.36 for stacked LSTM, 51.28 for GRU and 31.8 for vanilla LSTM GSO models. Similarly, a substantial difference in RMSE values was observed for a Test set of data such as 702.95 for SVR, 457.17 for RF, 447.5 for KNN, 211.03 for stacked LSTM, 211.05 for GRU and 38.85 for vanilla LSTM GSO. The least RMSE values were obtained in the case of the GSO LSTM model and there was not much variation in the training and test data values, which was existing in other models. In this study, a single CPU vanilla LTM model infused with GSO has been used. A parallel version of GSO could be applied successfully to enable parallelization. This would achieve scalability and efficiency.

Keywords: Glowworm Swarm Optimization Algorithm (GSO), Long Short-Term Memory (LSTM), Deep Learning, Machine Learning, Sliding Window Mechanism, Walk Forward Validation, Hyperparameter Tuning

Introduction

Financial engineering is a combination of various disciplines borrowed from finance, mathematical tools, and computer algorithms. The discipline of financial engineering is about predicting future trends, pattern discovery, and portfolio management with the key aim of reaping maximum profit for the organization. An application area of machine learning in financial engineering is forecasting stock market trends. This is

based on time series analysis that involves the study of stock prices over a long duration. Deep learning networks specifically, Long Short-Term Memory (LSTM) method can be applied to this since the problem is sequence prediction. The concept of machine learning applied to the financial domain is of utmost interest as the speed with which humans can solve problems is limited in nature. In such a scenario, a branch of machine learning has emerged which is called deep learning. The objective of the current research paper is to discuss the usage of LSTM, for predicting future stock prices and how this could be optimized further for better prediction results, with the help of the GSO algorithm to lower the error between predicted and actual prices. Deep learning network systems involve supreme computation of a humungous volume of data. Thus, it helps to overcome the limitation of the human brain's scalability. The performance of deep learning networks can be optimized by fine-tuning their hyperparameters. One of the premier choices for this purpose is Swarm algorithms. In this study, the focus is on deciding the optimal topology of LSTM by tuning parameters like the number of units, window size, batch size, epochs, and learning rate for training the LSTM model by fusing it with the GSO algorithm.

The LSTM models have been used in multiple ways in various earlier studies in the field of time series predictions. Some of the studies used it for feature selection and some others for prediction tasks. However, most of the studies focused on univariate or multivariate series with single-step forecasts. The study proposed is considering both univariate and multivariate series along with multistep prediction.

The optimization algorithms are categorized as evolutionary and swarm intelligence algorithms. GSO is one of the swarm intelligence algorithms particularly suited for the optimization of complex models since it is a multimodal and derivative-free optimization algorithm. GSO is proposed in the study to improve performance and time efficiency. The simple LSTM model may not handle the parameter selection problem efficiently and it may result in poor regression prediction ability of the model. Since the financial data is non-linear and unpredictable and it is difficult to build an accurate model to predict future prices, we can see that the LSTM model can make full use of the advantages in dealing with complex nonlinear problems and the long-term time-series data. With the increasing requirement of prediction accuracy and prediction time, the single prediction algorithm of LSTM has been unable to meet the needs, hence the GSO algorithm is used to optimize the LSTM network parameters and improve the accuracy of the prediction results. Therefore GSO-based LSTM prediction model, which uses the advantages of LSTM in processing long-term sequence prediction problems and optimizes the LSTM model parameters with the parameter optimization algorithm GSO is proposed.

Bouktif *et al*. (2018) present the domain of smart power grids. The accuracy of electric load forecasting was carried out by training various models and selecting the most appropriate of these. A genetic algorithm was used to fine-tune the topology of the LSTM in the study. The results showed decreased RMSE values with the usage of LSTM. Schaul *et al*. (2010) discusses a machine-learning library PyBrain, intended to provide powerful algorithms for machine-learning tasks. Seidy (2016), presents a study on PSO with the center of mass technique to train a model for the prediction of stock market price. Stajkowski *et al*. (2020) go on to illustrate forecasting real-time river water temperature using GA-optimized LSTMs. The GA model was used to forecast water temperature. Sagheer and Kotb (2019) used Time Series Forecasting (TSF) to predict future values with past data for a given sequence. For this purpose, the petroleum industry domain was chosen and the genetic algorithm was applied to optimize LSTM configuration. Krishnanand and Ghose (2005) introduce the concept of the glowworm metaphor in the field of collective Robotics. The firefly algorithms are used for tuning parameters of RNN in Ahyar *et al.* (2020). Thakkar and Chaudhari (2021) present a deep survey of Particle Swarm Optimization algorithms. Brownlee (2017) briefly explains the use of LSTMs in the field of Sequence Prediction using Python. Ghosh *et al*. (2019) calculated companies' net growth to predict the future growth of a company using the LSTM model. Yadav *et al*. (2020) used LSTM for the Indian stock market and gave considerably good prediction results. Yao *et al*. (2018) proposed an LSTM model for high-frequency stock trend prediction and observed that LSTMs can find hidden patterns in data. Stock market indices were predicted using the LSTM approach (Shen and Shafiq, 2020). Roondiwala *et al.* (2017) used four deep learning architectures to forecast the close price of stocks of NSE, India, and NYSE, New York. The model could predict both stock markets effectively. Hiransha *et al*. (2018) proposed trend prediction of stocks with the aid of deep learning networks. It resulted in high accuracy. Wang *et al*. (2020) apply the theory of mean-variance for selecting a portfolio by applying LSTM networks. The study was applied to UK stocks during 1994-2019. The Indian stock market is studied for optimizing the portfolio of stocks using k-means clustering (Kedia *et al*., 2018). Stocks from the Bombay Stock Exchange (BSE) are chosen for the study. In Yun *et al*. (2020) a two-stage deep learning approach is used for ETF assets. Chou and Nguyen (2018) study the effect of a hybrid model using the sliding window technique combining metaheuristics on the Taiwan stocks. The model outperforms other models in its prediction capabilities.

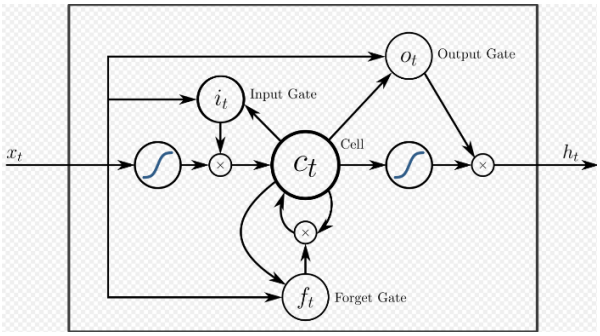

Fig. 1: An LSTM Cell; https://upload.wikimedia.org/wikipedia/commons/thumb/5/53/Peephole_Long_Short-Term_Memory.svg.png

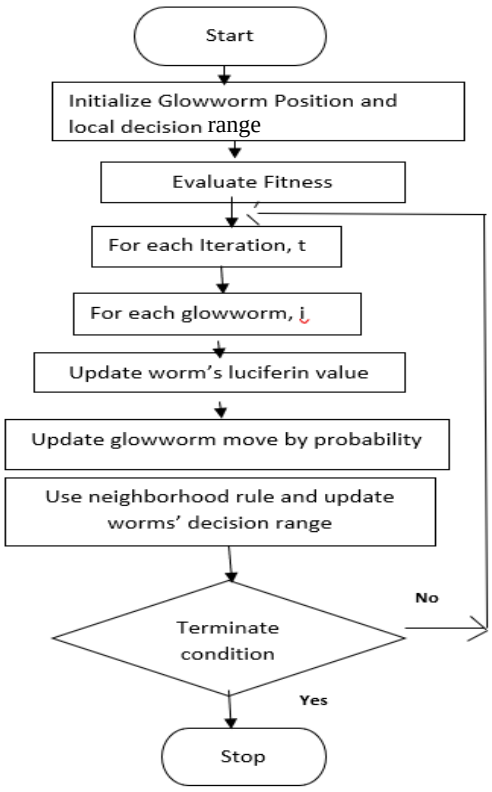

Fig. 2: Glowworm swarm optimization flowchart

Deep Neural Networks

Deep learning consists of numerous hidden layers as opposed to Simple MLPs with two or three hidden layers. The greatest benefit of deep learning is that it scales with more data. (More data + bigger models + more computation). With humongous datasets and the availability of computationally faster resources, deep learning took off very quickly and found a place in the domain of niche problem areas. However, the complexity of these networks and the abstraction of layers are difficult to comprehend. As a result, optimizing these networks using trial-and-error methods consumes enormous time. For the above reason, there is a tendency to use other approaches like metaheuristics and evolutionary algorithms to optimize these deep learning networks. Belonging to deep neural networks, a category called recurrent neural networks is used to process sequential data. There are three main types of RNNs: LSTMs, Bidirectional LSTM's and gated recurrent units. LSTMs remember short and long-term values. The information flow in LSTMs is regulated using three gates, input, output, and forget gates. DNNs have Fully Connected (FC) layers. The linear part is the activation layers. ReLu, logistic, sigmoid, and tangent-hyperbolic are the common activation functions. They are particularly useful in sequence prediction problems using time series data. The LSTM has become the focus of deep learning (Yu *et al*., 2019) Fig. 1 shows an LSTM cell model.

Glowworm Swarm Optimization (GSO) Algorithm

Genetic Algorithms belong to the family of evolutionary computation and GSO belongs to the family of metaheuristic algorithms (Eiben and Smith, 2015). The GSO algorithm was proposed by Kaipa and Ghose (2017). According to this metaphor, the glowworms attract each other in the night by emitting luciferin. The more luciferin a glowworm has, the more the emitted light and attracts more neighborhood glowworms to it. The result of each iteration of the algorithm gets better and finally, it reaches the optimum value of the solution. The steps of the GSO flowchart are illustrated in Fig. 2.

There are three different phases in glowworm swarm optimization. In Phase1 the concentration of luciferin for each glowworm is updated. The formula for each phase of the GSO is formulated by Krishnanand and Ghose (2009); Kaipa *et al*. (2017) as Eqs. (1-4).

$$Li(t+1) = (1-\rho)Li(t) + \gamma F(xi(t+1)) \quad (1)$$

Li (t) is the level of luciferin of glowworm i at time t, ρ is the luciferin decay constant ($0<\rho< 1$), γ is the luciferin enhancement constant, and F (xi (t)) represents the value of the objective function at glowworm if's location at time t. The second phase has two parts: Movement phase 1 and movement phase 2. In movement phase 1, we find the probability of movement for I to j:

$$pij(t) = \frac{lj(t) - li(t)}{\sum k \in Ni(t) \quad lk(t) - li(t)} \quad (2)$$

where, $j \in Ni(t)$, $Ni(t) = \{j: dij(t) < r\ di\ (t); li(t); li(t) < lj(t)\}$ is the set of neighborhoods of glowworm I at time t, $dij(t)$ represents the Euclidean distance between glowworm i and j at time (t). $rdi(t)$ represents the variable neighborhood range associated with glowworm i at time t. The variable is bounded by a radial sensor range

(0<*r di* < *rs*). Then, the discrete-time model of the glowworm movement can be stated by Eq. (3).

In movement phase 2 the movement from *i* location towards *j* location is calculated:

$$x(t+1) = x(t) + st\left(\frac{x_y(t) - x_i(t)}{\left\|x_j(t) - x_i(t)\right\|}\right) \quad (3)$$

Then $xi(t) \in R^m$ is the location of i^{th} glowworm, at time *t*, in the *m*-dimensional real space R^m. $\| \|$ represents the Euclidean norm operator and (*st*>0) is the step size.

In phase 3 also the range of neighborhood locality is updated by applying the following Eq. (4):

$$r^i(t+1) = \min\left\{r_s, \max\left\{0, r^i(t) + \beta(n_t - \left|N_i(t)\right|)\right\}\right\} \quad (4)$$

where, β is a constant, n_t indicates a number of neighbors and rd denotes the sensory radius, and $N_i(t)$ the neighborhood range.

GSO Infused LSTM Forecast Model

The parameters of a deep neural network say for example LSTM is difficult to be configured manually. To reduce the time taken for tuning these parameters we use the glowworm Swarm optimization algorithm. Unless the hyperparameters of the LSTM network are tuned properly, we may not get good prediction results. The GSO algorithm works on the like of glowworms in nature. The glowworms have a property of emitting light called fluorescein. With the emission of this light, the glowworms try to gather around each other. Each glowworm will be having a location. The location of each glowworm represents a potential solution. This location keeps on changing as the glowworms tend to gather around a new location based on the strength of the light emitted by its neighboring glowworms. This updating of location is based on a probabilistic strategy. Gradually, the majority of the glowworms tend to assemble at the same location based on the concentration of the light emitted. The iterations of the algorithm continue until this optimum location is reached. In the algorithm proposed, the initial location is denoted by Lo. Lo is given as (window size, batch size, iterations, epoch no, learning rate). These arguments denote the hyper-parameters of the LSTM model that are to be tuned. After the set number of iterations, the GSO outputs the best combination of parameters for the LSTM. The fitness function is taken as the RMSE calculation.

The parameters of the GSO are Lo, *Ni*, and *J*. To find the optimum location, these parameters are continually modified in the iterations, until the optimum value of the Lo is reached. The function to be considered is the RMSE function which yields the minimum value. Corresponding to each RMSE value in the respective iterations, the hyperparameters of LSTM are generated and output, which are the window size, batch size, epochs, and learning rate.

The steps governing the GSO fused LSTM Algorithm are reproduced below:

1. Initializing glowworm positions and local decision range
2. The Luciferin value of i^{th} glowworm is updated by the rule given by Eq. (1)
3. Probability of finding neighbors. Applying Eq. (2) this is calculated
4. Movement from an existing location to a new location, wherein movement from *i* location towards *j* location is calculated as given by Eq. (3)
5. Update the location of glowworm *i*. Use update phase Eq. (4)
6. Output optimal parameters: The fitness function uses the RMSE value of the LSTM prediction model
7. End

Pseudo-code of GSO is presented below:

Algorithm: Parameter optimization by GSO

Input: Parameter values of the GSO algorithm,
Range of Model Parameters

Output: The model's optimal hyperparameter values

```
1.  Initialize parameters of GSO
2.  n = 1
3.  While n ≤ t
      do place random glowworms;
4.  i = 1
      While i ≤ N
        Do li = updateLuciferin ();
           Ni = find Neighbor ();
             J = maxProbability(i);
             Update Location(i);
             UpdateLuciferin(i);
        end
    If RMSE is satisfied then break; end
    end
```

Proposed Methodology with Dataset Specifics

The proposed methodology with the implementation procedure for fine-tuned model building is shown in Fig. 3. The GSO fused LSTM model is used in the implementation procedure as mentioned in the pseudocode.

The pseudo-code for the GSO fused LSTM model is elaborated as given.

Table 1: NSE sector dataset

Symbol	Date count	Date min	Date max
Asian paint	5497	2000-01-03	2021-01-29
Britannia	5497	2000-01-03	2021-01-29
Hindustan lever	5497	2000-01-03	2021-01-29
ITC	5497	2000-01-03	2021-01-29
Titan	5497	2000-01-03	2021-01-29

Table 2: BSE sector dataset

Symbol	Date count	Date min	Date max
Asian paint	5495	2000-01-03	2021-01-29
Britannia	5495	2000-01-03	2021-01-29
Hindustan lever	5495	2000-01-03	2021-01-29
ITC	5495	2000-01-03	2021-01-29
Titan	5495	2000-01-03	2021-01-29

Algorithm: Optimized LSTM predicting stock price

Input: Open, High, Low Prices
Output: Predicted close Prices, RMSE
1. Start
2. Initialization: Input data of multiple stocks
3. Do Data Pre-processing
4. Do Feature Selection
5. Do Normalization
6. Split into Training, Test, and Validation sets
7. Shape the inputs into a tensor
8. InitParameters (); //Initialize parameters in the GSO algorithm
9. Get the parameter combination from algorithm1;
10. $i = 1$
while $i \leq N$ do LSTM. Train;
11. Compile the model with GSO-parameters
12. while $N < i <$ max do
Li <- predict_price ();//predict future close price
Stock add (Li);
Check with actual value
Output validation RMSE
13. i++;
14. End for
15. Termination: Stop

Dataset Details

The dataset collected from two different sources is used in this study. The details of the dataset with their available source are mentioned as follows.

Dataset 1: Stocks from NSE-sector consumer goods, duration: 03-01-2000 to 29-01-2021 (collected from the source: https://www.nseindia.com/report-detail/eq_security).

Here, each stock has 5497 records and the total number of records in the NSE dataset is 5497×5 = 27485 records. The training dataset is used ranging from 2000-2017 and the test dataset ranged from 2018-2021.

The stock details from NSE consumables sector with the duration are shown in Table 1.

Dataset 2: Stocks from BSE-sector FMCG, duration 03-01-2000 to 29-01-2021.

(Collected from the source: https://www.bseindia.com/markets/equity/EQReports/StockPrcHistori.html?flag=0).

The stock details from BSE Sector FMCG with the duration are shown in Table 2.

In this dataset, each stock has 5495 records and the total number of records in the BSE dataset is 5495×5 = 27,475 records. The training dataset contents ranged from 2000-2017 and the test dataset ranged from 2018-2021.

Stock market data from NSE was used, for which 5 companies were chosen. The stock data of the selected companies was obtained from the national stock exchange of India. These were categorized according to the sector to which they belong, which is the consumer goods sector.

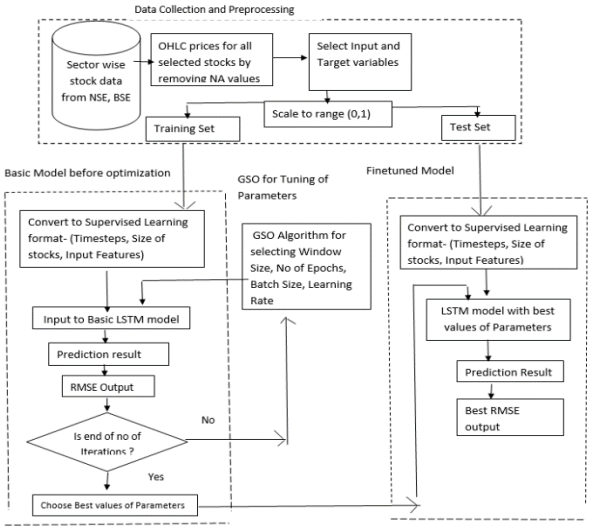

Fig. 3: Proposed methodology

Fig. 4: Close prices of selected stocks

Table 3: Machine models with their hyperparameters used against GSO LSTM

Algorithm	Hyperparameters
SVR	kerne l = 'rbf', C = 0.001, gamma = 0.1, degree = 3, epsilon = 0.1
RF	n_estimators = 100, random_state = 0
KNN	n_neighbors =15, metric = minkowski
LSTM	Loss =' mse', optimizer ='adam', Three lstm layers with 32 nodes
GRU	Loss =' mse', optimizer ='adam', four GRU layers with 32 nodes
LSTM GSO	Loss =' mse', optimizer ='adam', single lstm layer with 16 nodes

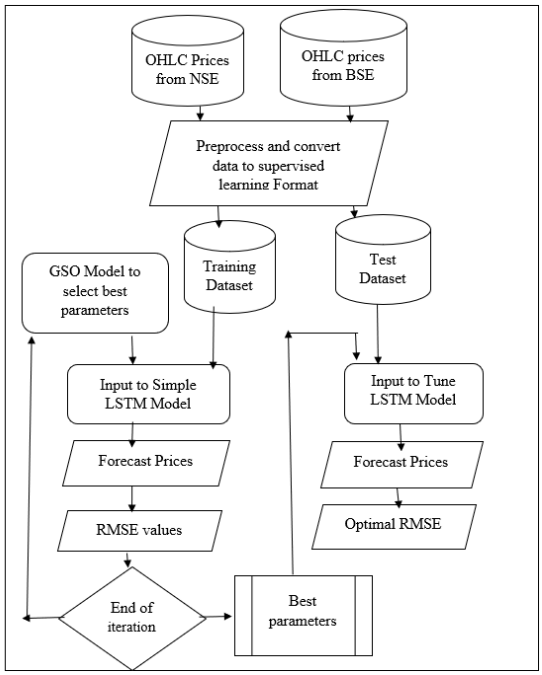

Fig. 5: Workflow of methodology

The stock closing prices from these five stock companies, which also belong to the NIFTY 50 index, for the period 03-01-2000-29-01-2021 are contained in the dataset. The stocks selected were ITC and Titan, Britannia, Asian paints, and Hindustan Lever. These stocks belong to the sector of Consumer goods of NSE.

In the same manner, the same set of stocks was selected from the FMCG sector from the Bombay stock exchange (S & P -BSE Sensex). This data is also for the period 03-01-2000-29-01-2021. The set of data so obtained helps to compare the performance of the stocks of the two trading exchanges in the Indian stock market. Using Nifty 50 stocks from national stock exchange India an LSTM was built for selected stocks. Later it was used as a generic model for the same set of stocks selected from the Bombay stock exchange. Figure 4 shows the plot with the close prices of three stocks selected from BSE Sensex, FMCG sector Britannia, Hindustan lever, and ITC.

Even though we have taken 2000-2017 as the training dataset, we can even consider a 5-year period as a suitably sufficient size of the training dataset. However, the more the period for the training set, the model can capture any seasonal fluctuations that reoccur.

Implementation Procedure

In this study we choose the 5 stocks which are selected as mentioned in the previous section; they are Asian paints, Britannia, Hindustan lever, ITC, and Titan. All the companies' data is not available for all the days. The companies which have one or two days of repeated values will be preprocessed so that the duplicate values are removed. The date count is checked for all five companies and the minimum and maximum dates are checked. The min and max dates for all the companies are set to be uniform. The timeseries are checked for stationarity. The next step is deciding upon the Input and Target features. This part of the study consists of the following subsections:

1. Univariate series, sliding window mechanism for single-step prediction
2. Multivariate series, sliding window mechanism for single-step prediction
   a GSO Tuned LSTM Model using OHLC Prices + 18 Technical Indicators which are derived from OHLC prices
   b Feature importance to select best 4 features
   c GSO Tuned LSTM using OHLC prices as independent features
3. Multivariate series, sliding WINDOW for multistep prediction
   a GSO Tuned LSTM using OHLC prices as independent features, with future predictions for multiple days parallelly

Univariate Series, Sliding Window Mechanism for Single Step Prediction

As the experiment is carried out using different algorithms, a univariate series is used, wherein the close Price of the past period is used to forecast close prices for future days. A walk-forward validation is applied and the data is split into train and test data and then into the supervised format required by LSTM models. The following Table 3 shows the different models that were used for the prediction of the data for the selected stocks. The above-mentioned naïve algorithms were used for the comparison of RMSE with each other. The implementation of the GSO algorithm with LSTM for stock price prediction is the proposed model as given in Fig. 5.

Once the input values are finalized, the data is scaled to the 0, 1 range. The different prices may be measured at different scales. As a result, they do not contribute equally to the model fitting and there is a possibility of bias. To avoid this, the min-max scaler from Python's sci-kit library was applied. The scaling allows converting the prices into the range between 0 and 1 and thereby making the convergence of the algorithm easier. The stocks were divided into training and test sets with a walk-forward validation procedure, after which data was converted into a time series format. Each stock was fitted with the LSTM model iteratively. The fitting of each iteration with the RMSE result was captured and compared. The best values for different parameters like the number of neurons, window size, batch size, epochs, and LRATE were considered. The use of the LSTM model requires input data to be in a particular format represented as samples, steps, and features. Once the testing data is ready in this format, we feed it into the LSTM layer. The data is initially loaded into the LSTM. The hyper-parameters of the LSTM model namely Window size, epochs, batch size, and learning rate are generated as different solutions for different iterations of the GSO algorithm which is fused with the LSTM. These are the input values for those parameters in the LSTM. For each iteration, a unique set of values of these parameters is fed into the LSTM and the model gives the predictions. The RMSE value, that was computed is a measure of the deviations of these predictions from the actual values of the close prices of stocks. The fitness function of the GSO algorithm is designed to be the calculation of RMSE values. The lesser the RMSE value for a glowworm, the better the fitness. Thus, for various iterations of the GSO algorithm, the LSTM model is trained and the values of the different iterations become in turn the different values of the hyper-parameters, which are input to the LSTM. This process is carried out with a training set of data and the value of the individual with the least RMSE is selected. With the availability of the best hyper-parameter values, the model is fitted for test data and then predictions are made. The following values are passed to the initialization of hyperparameters of the LSTM model as recorded in Table 4.

The GSO-LSTM model is compared against other states of the art models, as well as the traditional Stacked LSTM model without any parameter tuning as shown in Table 5.

Table 4: Hyper-parameters of the LSTM

Hyper-parameters of GSO-LSTM	Range
No of nodes	10-50
Size of window	1-60
Batch size	1-100
Rate of learning, Lr	0.0001-0.1
Number of epochs	15-100

Finally, the GSO-LSTM was trained to take the RMSE as the metric for the fitness function. The least RMSE gave the best parameters. However, after the application of the GSO algorithm, the optimal parameters were obtained. The parameters that were tuned were window size, learning rate, Number of epochs, and batch size. Once the optimal values were obtained a two-fold walk-forward validation was done and the model was fit. The RMSE values obtained with the optimal parameters are as follows: For the stock Titan, the GSO-LSTM model gave an RMSE of 20.86 which was the least among all the compared models. In a similar fashion, The RMSE for the stock ITC was the least for the GSO-LSTM model which was 6.37. In all the cases the GSO-optimized LSTM showed a much lesser RMSE value than the other three models. Hence the proposed GSO-LSTM is performing better than similar prediction algorithms. In addition, the MAE metric values were also compared for the three models. These values also show a superior prediction performance by the GSO-LSTM model. With the optimal solution, the test data is fit to the model and predictions are made. The loss function for different epochs is shown for the two companies which were initially selected for training the LSTM. In the case of the GSO-LSTM model, for the two stocks selected, the trace of the losses for Training and validation sets of data shows that by epoch 5 the loss drops and stays more or less the same for the remaining number of epochs. This is shown for the stock ITC and Titan in Figs. 6-7.

Multivariate Series, Sliding Window Mechanism for Single Step Prediction

Feature Importance

Since a comparison between different models was made on a univariate series, a need arises to check whether the tuned model will give optimum results when more than 1 feature is chosen as input. In order to find out the impact that different features will have on the target feature namely the close price, a few features have been engineered as follows.

The original prices OHLLC are denoted as Open, High, low, last, and close prices and other technical indicators derived from these prices. Table 6 shows the technical Indicators used for multivariate series.

The fused GSO-LSTM model which gave the best parameters is used to fit the multivariate series. There were 18 derived features fed into the LSTM as input features against the target Close price. The sliding window mechanism is used to determine the steps which are the number of input days to be used for prediction. The results obtained for 3 stocks deviated much from the error metric used in the Univariate series. Table 7 lists the RMSE values while taking Univariate Series and then using multivariate series.

Table 5: Comparison of LSTM GSO with other models based on RMSE metric

Stocks		SVR	RF	KNN	Stacked LSTM	GRU	Vanilla LSTM GSO
ITC	Train	70.0500	13.9800	54.0500	33.1500	31.740	4.6300
	Test	80.7600	13.1400	11.6300	7.0400	10.920	6.3700
Titan	Train	302.5100	30.8800	127.8400	46.3700	61.190	21.7200
	Test	182.9300	21.1800	31.2700	15.7700	43.990	20.8600
Britannia	Train	436.7900	14.1000	53.0000	34.7300	33.130	68.8800
	Test	2664.6200	1825.0600	1726.3500	913.4600	899.970	110.3700
Asian paints	Train	283.0700	29.0500	141.0000	64.9700	89.640	27.5700
	Test	323.5300	33.0100	42.9700	26.7000	37.270	27.8000
Hindustan lever	Train	258.4500	17.7900	77.7100	42.6100	40.720	36.6100
	Test	262.9000	393.4700	425.2700	92.2000	63.090	28.8500

Table 6: Technical Indicators used for multivariate series

'open_1', 'high_1', 'low_1', 'last_1', 'close_1'	'Open'. Shift (1) and so on. The prices are shifted to one day
'ma7', 'ma21', and 'ma365'	Average prices, which is the rolling mean of the Close price for windows = 5,21 and 252 days These are the moving averages
'MACD'	Obtained from an exponential moving average of close prices for a span of 26 and 12 days
'Bollinger bands'	a) Obtained by adding the moving average for 21 days summed to 20 days
a) Upper band	rolling standard deviation *2
b) Lower band	b) Obtained by subtracting the moving average for 21 days from 20 days rolling standard deviation *2
'ema'	Exponential moving average of close prices for a span of 20 days

Table 7: RMSE values of the GSO-LSTM model with multivariate and univariate series

	RMSE	
Stocks	Train data	Test data
ITC-TechInd	4.4910	7.2890
ITC_Close	4.6350	6.3680
TITAN_TechInd	21.7880	23.7970
TITAN_Close	21.7200	20.8580
BRITANNIA_TechInd	74.1270	121.6320
BRITANNIA_Close	68.8750	110.3720
ASIANPAINT_TechInd	36.3130	45.6450
ASIANPAINT_Close	36.6050	28.8550
HINDLEVER_TechInd	30.0400	36.8590
HINDLEVER_Close	27.5660	27.8030

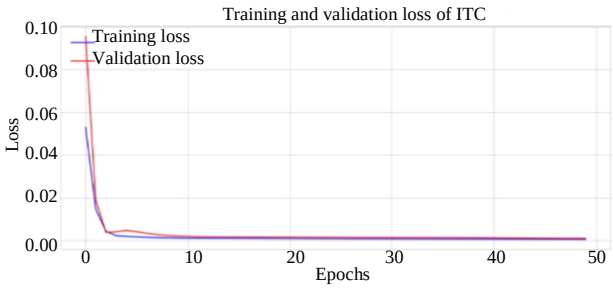

Fig. 6: Mean squared loss against epochs for stock ITC

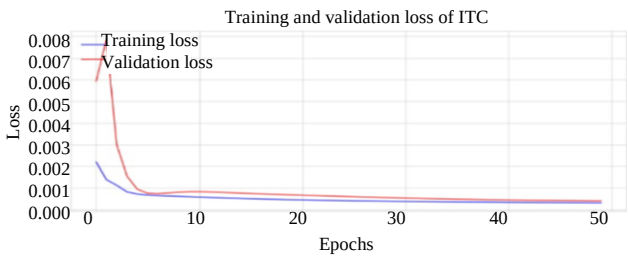

Fig. 7: Mean squared loss against epochs for stock TITAN

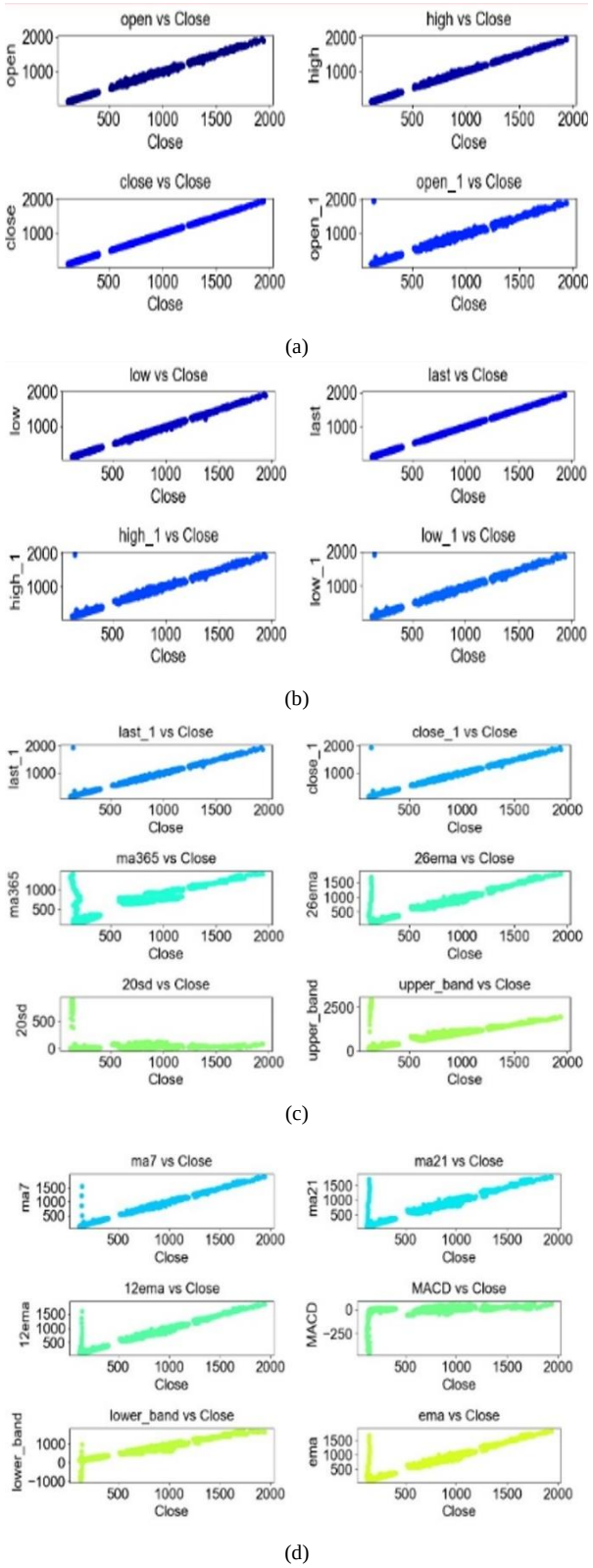

Fig. 8: (a) Relationship between close and other features; (b) Relationship between close and other features; (c) Relationship between close and other features; (d) Relationship between close and other features

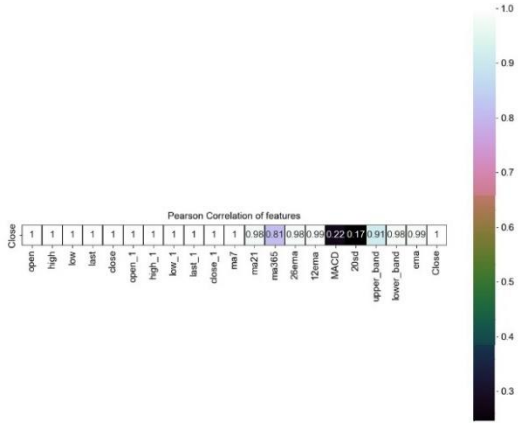

Fig. 9: Correlation of different features with close

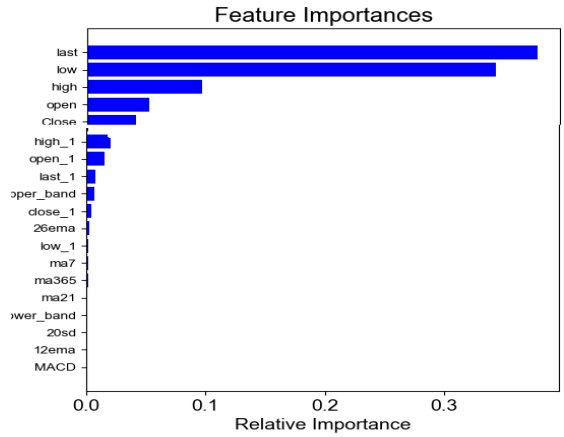

Fig. 10: Features according to importance generated by random forest model

It can be seen that there is not much value addition by including some important technical indicators. Hence a need arises to figure out the top-ranking features among all the features selected for the study. To determine feature importance, a random forest model is used. The features are fed into the RF regressor and using the feature importance parameter after fitting the data to the model the top 4 ranking features that impact the close prices of future days are determined. The correlation matrix is also used to aid in this purpose. The relation between the features is shown in Fig. 8.

From the diagrams above there is a clear-cut picture that all the OHLLC prices and then open_1, high_1, low_1, last_1 and close_1 prices is showing a straight line with the close price. The moving average, ema the Bollinger bands, and MACD figures are not exactly straight lines.

Pearson's correlation matrix also stresses the same concept and the correlation of the features is shown in Fig. 9.

To decide on the top 5 features the feature importance graph enables us to do that. The feature importance generated through the random forest is shown in Fig. 10.

Table 8: RMSE, MSE, and MAE with all three scenarios

		RMSE		MSE		MAE	
	Stocks	Train	Test	Train	Test	Train	Test
Multivariate Series with technical indicators	ITC-TechInd	4.4910	7.28900	20.1690	53.126000	1.4160	5.0320
Univariate series with close prices	ITC_Close	4.6350	6.36800	21.4800	40.548000	1.6800	3.9950
Multivariate series with OHLLC prices	ITC_OHLL	4.3350	6.06800	18.7910	36.820000	1.2790	3.7940
Multivariate series with technical indicators	TITAN_TechInd	21.7880	23.79700	474.7140	566.280000	6.9700	16.5910
Univariate series with close prices	TITAN_Close	21.7200	20.85800	471.7730	435.046000	9.2920	14.2430
Multivariate series with OHLLC prices	TITAN_OHLL	20.1550	21.99200	406.2390	483.669000	5.3330	14.2230
Multivariate series with technical indicators	Britannia_TechInd	74.1270	121.63200	5494.8360	14794.449000	43.1590	73.6040
Univariate series with close prices	Britannia_Close	68.8750	110.37200	4743.8270	12181.984000	35.6980	59.8560
Multivariate series with OHLLC prices	BRITANNIA_OHLL	63.1620	106.44700	3989.3990	11331.036000	29.5880	54.6100
Multivariate series with technical indicators	HINDLEVER_TechInd	30.0400	36.85900	902.3770	1358.577000	25.8330	29.5150
Univariate series with close prices	HINDLEVER_Close	27.5660	27.80300	759.8710	773.027000	10.6000	19.2600
Multivariate series with OHLLC prices	HINDLEVER_OHLL	24.8810	30.08600	619.0780	905.188000	5.4070	19.9780
Multivariate series with technical indicators	ASIANPAINT_TechInd	36.313	45.64500	1318.6240	2083.447000	12.5820	38.9430
Univariate series with close prices	ASIANPAINT_Close	36.6050	28.85500	1339.9570	832.608000	14.2820	21.0340
Multivariate series with OHLLC prices	ASIANPAINT_OHLL	33.7410	28.38700	1138.4520	805.831000	9.2800	19.4120

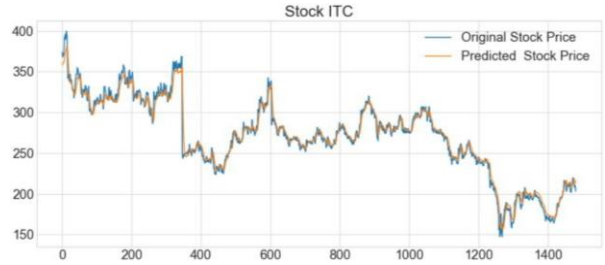

Fig. 11: Forecasted and actual prices of stock ITC

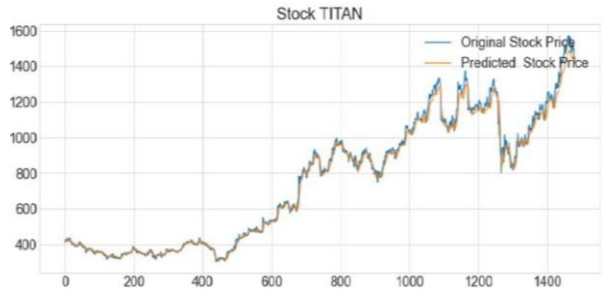

Fig. 12: Forecasted and actual prices of stock TITAN

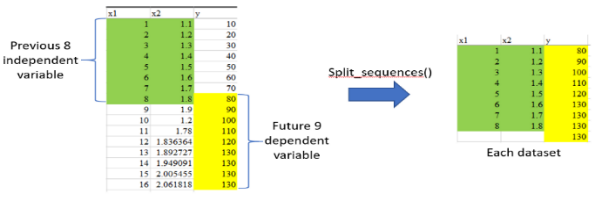

Fig. 13: Format of data split based on window size and lags

The graph shows that the Last price has the highest impact on the close price which is true as per the real market trend. This is followed by low, high, open, and close prices. Hence the decision is to take out only these features and run the GSO LSTM model to verify the results.

Multivariate Series, with OHLLC Prices as Input and Close as Target for Single Step Prediction

The experiment was then repeated for the same set of parameters of LSTM obtained from the GSO algorithm, with the first 5 important features. The results were studied for these features. It turns out that the GSO LSTM is further optimized with the selected set of OHLLC prices as Input features. The results of the model on train and test data for the selected stocks are shown in Table 8.

A graph showing the trace of original and predicted values helps to clarify the fact that with the drop of RMSE, the predicted line comes closer to the plot of actual values. Figures 11-12 show the graph of original and predicted close prices after fitting the GSO LSTM model.

A total of 18 features were input to the GSO LSTM model. However, from the table of results, we can very well see that the addition of these features did not enhance the performance metrics. Hence it was decided to keep only the top-ranking original features, namely OHLL to predict the close prices of the future.

Multivariate Series, with OHLLC Prices as Input and Close as Target for Multistep Prediction

In Time-series prediction the future target variable (y) is forecast using the values of the independent feature (s)(x). When a single independent feature predicts the target value, we get forecasting that is univariate. As opposed to univariate series when multiple × variables are used to predict the y value and if the prediction is only for the immediately occurring time slot it is referred to as a one-step forecast. A slight

modification in this approach where multiple steps ahead of time are predicted using multiple independent variables results in a multi-step forecast.

In this section, the model which is the optimized LSTM GSO model will do multistep forecasting with multivariate series.

To illustrate this, if the steps in (window_size) = 8 and steps_out (days to forecast) = 9, the data will be arranged in Fig. 13, keeping 8 days of each independent variable and the next 9 dependent values.

Hence if we look back on 8 past observations of (X_1, X_2, X_3, X_4, X_5) 5 independent variables, to forecast 10 days multistep ahead, the size of X and y would be (No of rows, 8, 5) and (No of rows, 10). The array in general will be arranged with rows of data followed by window length and number of independent values for X and rows of data followed by time lag for y respectively. Therefore, in the GSO LSTM model, a single prediction for the immediate future day is done by retaining the features of the window length period. This is achieved with what is known as a single-shot model, where the complete series of predictions is done in a single step. This method is also called a direct method. This can be implemented by setting the dense layer with the number of output units equivalent to steps_out x no of features in Fig. 14.

The following parameters were used to produce RMSE values for multistep future predictions for the selected stocks. A window size of 57 and lags out for a period of 15 days produced the following RMSE values in Table 9.

The various multistep prediction results are shown in the following charts from Figs.15-19.

Comparison of Proposed Work with Existing Studies

The proposed work with its implementation procedure is compared with existing studies as mentioned below. Wang *et al*. (2020) used the LSTM model in conjunction with asset preselection. They then integrated the Mean-Variance Model and applied the model to the UK stock exchange 100 index. The data used was from March 1994-March 2019. The average RMSE obtained was 0.0543. However, in our study, we have tuned the LSTM model so that it will converge to the optimal point faster using the GSO algorithm. Kedia *et al*. (2018) used k-means clustering for forming a stock portfolio using the BSE 100 stocks. They compared the rate of return of these stocks to the Benchmark of the Indian stock exchange. In our study, the stocks have been compared to the nifty 50 index and are outperforming the Index. Roondiwala *et al*. (2017) studied the forecasting of Indian stocks using the LSTM model. They used the nifty 50 from the NSE for the period 01.01.2011-Dec 2016. The window size used was 22 days. They obtained RMSE values for single variate and multivariate data ranging from 0.01491-0.00983 for the model. In our study, the window size is one of the parameters that is tuned by the GSO and the optimal value is found based on the minimum RMSE value. Ghosh *et al*. (2019) used the LSTM model to predict the Price & Company Growth of stocks. In the model proposed by us, the LSTM Model is used to forecast Returns. A novel algorithm PSOCOM was proposed by Seidy (2016) to study data from the NASDAQ-100, DJIA, and S & P 500. The duration of data was from January 2, 2005-December 31, 2014. The mape values were calculated and the MAPE of long-term prediction for the DJIA index is equal to 0.8601% in that study. In the GSO LSTM model, the GSO algorithm optimizes the performance of the LSTM model and returns a fitness value for the least RMSE. Therefore, in comparison with the earlier studies the GSO LSTM model has resulted in a lower RMSE for the NSE stocks.

Table 9: RMSE values for GSO LSTM Model for a single day and for the entire 15 steps

	RMSE	
Stocks	Day 1	Day 1-15
Titan	75.0950	93.2320
ITC	63.6960	32.6390
Britannia	336.9280	219.8640
Asian paints	122.3790	100.8770
Hindustan lever	141.8740	488.1130

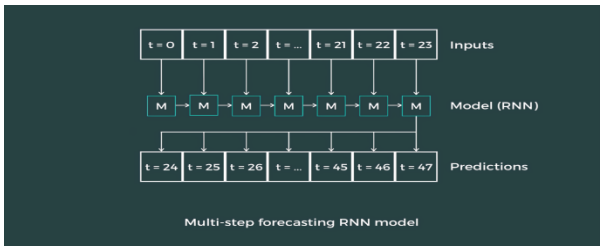

Fig. 14: GSO LSTM model with input and output lags

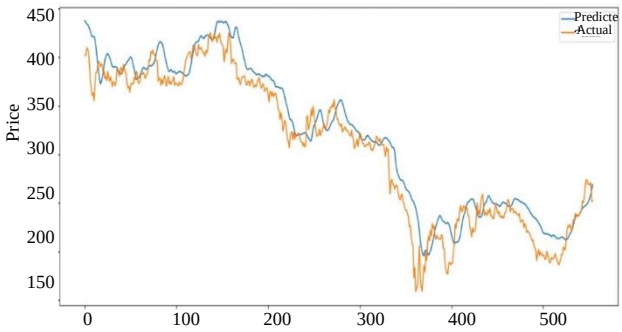

Fig. 15: Forecasted and actual prices of stock ITC for multistep prediction

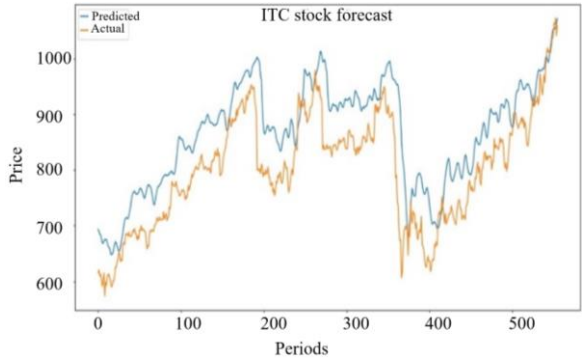

Fig. 16: Forecasted and actual prices of stock TITAN for multistep prediction

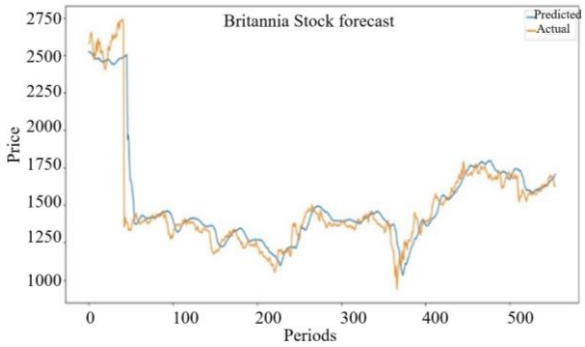

Fig. 17: Forecasted and actual prices of stock Britannia for multistep prediction

Results

The reason for choosing LSTM in this study is that is observed that LSTM is good for time series predictions as it can memorize past values. LSTM can capture the non-Linearity of time series data in the stock market. The data with overfitting and underfitting can be removed using GSO fusion. Early convergence using GSO fusion is achieved in this study. This procedure minimizes Prediction error and it lowers RMSE values when compared to other models. The low values of these loss function indicators show that the model is efficient in predicting stock closing prices.

We have chosen RMSE as the metric determining the fitness value of GSO. RMSE or Root Mean Square Error represents how much-predicted value differs from the actual values in a prediction /forecasting problem. The equation for calculating RMSE is given in Eq. (5):

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(predicted_i - actual_i)^2}{N}} \quad (5)$$

Here, Predicted$_i$ stands for the predicted value and Actual$_i$ represents ground truth, and N is the number of observations. There are two purposes for using RMSE in this study. It is used as a metric for training the LSTM,

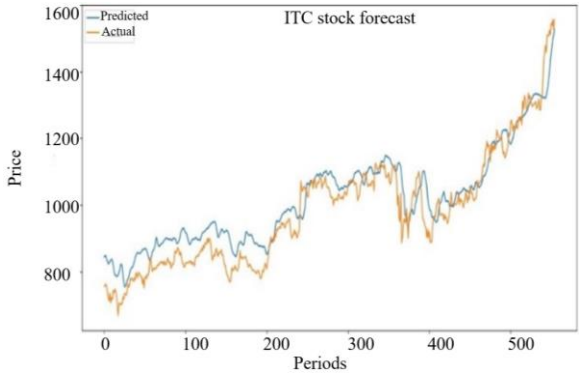

Fig. 18: Forecasted and actual prices of stock Asian paint for multistep prediction

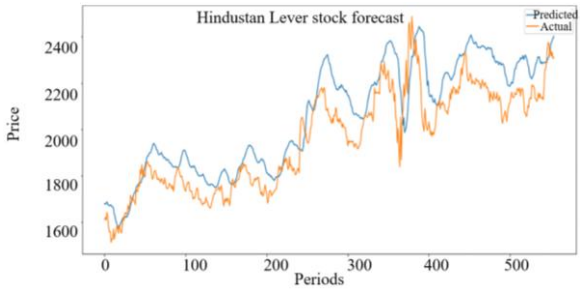

Fig. 19: Forecasted and actual prices of stock Hindustan lever for multistep prediction

such that it helps us to decrease the error with each iteration. Based on whether RMSE is small or large, we can evaluate trained models for usefulness /accuracy. The GSO algorithm-tuned LSTM generated various combinations of hyper-parameters for each case. The best model with the least RMSE values is selected and generalized for other stocks too. For those stocks too, the model resulted in lower RMSEs compared to the LSTM without any optimization. Referring to the table of comparison among various models, we can see that the model GSO LSTM which is proposed in the study outperforms other models in terms of the Loss function. The model results in the least RMSE value for the selected stocks. We can see that by tweaking the parameters in our model, we can bring the RMSE below a certain threshold. However, there is no predetermined threshold for “small enough RMSE”. After tweaking the parameters, an optimized model was found with the lowest RMSE. However, given the same time period, the LSTM GSO model invariably outperforms all the other models taken for study.

Statistical Significance

To compare the significance of the model prediction F-Test was carried out. The Random Forest, KNN, support vector Regressor, Stacked LSTM, and GRU models using Close Price were tested with the tuned

GSO LSTM model using OHLL along with close price. All other models were treated as restricted models whereas the GSO LSTM model with OHLLC was considered to be the full model. The F-test results show that the GSO LSTM model fits the data with lesser errors. This proves the supremacy of the selected model.

For the Training set of data, the mean RMSE values for the different models were obtained as 270.17 for SVR, 21.16 for the random forest, 90.72 for KNN, 44.36 for stacked LSTM, 51.28 for GRU and 31.8 for vanilla-LSTM GSO models. Similarly, a substantial difference in RMSE Values was observed for a Test set of data such as 702.95 for SVR, 457.17 for RF, 447.5 for KNN, 211.03 for Stacked LSTM, 211.05 for GRU and 38.85 for Vanilla LSTM GSO. The least RMSE Values were obtained in the case of the GSO LSTM model and there was not much variation in the training and test data values, which was existing in other models.

Discussion

The comparative analysis of the proposed study with the existing studies is compared by considering various aspects like dataset, features, and various performance measures. The comprehensive performance evaluation is shown in Table 10.

The loss function test results show that the MSE, MAE, and RMSE of the GSO LSTM hybrid model are smaller. A good R-squared value explains the strength of the prediction of the model. Values nearer to 1 are considered good. R squared values for the LSTM GSO model are more than 0.95 which proves the supremacy of the model. The RMSE values also show a considerable decrease in the optimized model. Moreover, instead of going for complex architecture a simple single-layer model could forecast with the least RMSE. Also, the model is developed by using walk-forward validation on the dataset which is essential for a time series problem.

Table 10: Comparative analysis and evaluation of the proposed study

Basis of comparison	Proposed study	Existing study 1 Wang *et al*. (2020)	Existing study 2 Gao *et al*. (2021)	Existing study 3 Xiao and Su (2022)
Model name	GSO fused vanilla LSTM model	LSTM Model with Asset Preselection	Optimized LSTM and GRU models	ARIMA-LSTM model
Dataset duration and source	Duration: 03-01- 2000 to 29-01-20215497 days NSE and BSE stocks from consumer goods sector of The NIFTY 50 Index	March 1994 to March 2019, to August 3, 2021, the UK stock exchange 100 index	April 11, 2007, 3,481 days	2010-2019 at the New York stock exchange, 55,875 sets of time series data was trained and each set has 24-time steps
Dataset of stocks used	Stocks used: ITC, Titan, Britannia, Asian paints, and Hindustan lever No of records: 5497* 5stocks = 27,485 from NSE5495*5 stocks = 27,475 from BSE	21 stocks from FTSE 100	Data of the Shanghai composite index (000001)	150 stocks
Features	Open, high, low, last and close prices Used only technical indicators that are ranked top 5 using the random forest recursive algorithm	Adjusted open prices, close prices, the highest prices, the lowest prices and the trading volume of assets	Open price, highest price, lowest price, trading volume and other common technical indicators, such as OBV, KDJ, BIAS, RSI, CCI and MFI, as well as other stock price judgment technical indicators and PSY indicators reflecting investors' psychological mood	-
MSE	Mean Train MSE: 1447.972 Mean Test MSE: 3112.109 Least MSE(ITC): Train MSE: 21.48 Test MSE: 40.548	Mean MSE: 0.0033 for LSTM model	Train MSE: 816.3383 Test MSE: 1199.1475	-
RMSE	Least RMSE Train for ITC:4.63 Test for ITC: 6.37 Mean RMSE Train: 31.4956 Test: 41.4972	Mean RMSE 0.0543	Train RMSE: 28.5716 Test RMSE: 34.6287	-
MAE	Mean MAE Train: 14.15993 Test: 26.27267 At least MAE (ITC) Train:1.68 Test: 3.995	Mean MAE of 0.0303	Train MAE:20.2925 Test MAE:20.1759	-

Table 10: Continue

R squared	R squared: 0.9974 0.9967	R squared: 0.2621	NA	NA
Return prices	NA	NA	NA	Average return price of the LSTM prediction model is 14.01
Architecture	No of neurons-16, Single-layer, Learning rate = 0.003 Epochs-200. Obtained after optimization from the GSO	NA	Number of neuron layers are set to 2 and 3, the number of neurons is set to 8, 16, and 32, the learning rate is usually set to 0.001 and the number of iterations is set to 1000	NA

Conclusion

The study compared the results of the GSO LSTM model with other naïve regressors like support vector regressor, random forest regressor, k-nearest neighbor, Stacked LSTM, and GRU models. Upon comparison, the RMSE values could be considerably reduced in the vanilla-LSTM GSO model. This improved prediction accuracy is a significant contribution to this study. In addition, the time taken to attain the optimal parameters was reduced considerably using the hybrid model. By minimizing the error in the predicted and actual values of stock prices by integrating the glowworm swarm optimization algorithm into the selected LSTM Model, it is possible to optimize the prediction results of LSTMs. With this objective, the Indian stock market data from NSE (nifty 50) and BSE is used as an illustrative example for model training and evaluation, even though several optimization methods are available to fine-tune LSTMs, generally, they consume a lot of time and are resource intensive. The use of the GSO algorithm to achieve optimization could be justified in this context. In the current study, therefore, a simple GSO algorithm approach, which is used for the hyper-parameter tuning of a basic LSTM model helps in improving the efficiency of prediction results without following an exhaustive search.

The empirical results suggest that the proposed method minimizes the difference in predicted and actual values of stock market prices, in comparison with alternative approaches employed in the study. The study implies that the prediction error of LSTMs can be further lowered by the appropriate selection of optimal parameters for the model.

Based on the GSO LSTM model, in the future, a new automated system can be developed to select stocks with the highest profit and minimum risk. In this study, a single CPU vanilla LTM model infused with GSO has been used. A parallel version of GSO could be applied successfully to enable parallelization. This would achieve scalability and efficiency. The parallel version of GSO-infused LSTM makes it easier to handle growing datasets. With this aim, we could utilize multiple processor nodes. In the proposed study the parameters considered for tuning were window size, epochs, learning rate, and batch size. The study can be further extended to consider other parameters like adding more layers, weight initialization, and drop-outs.

Acknowledgment

Thank you to the publisher for their support in the publication of this research article. We are grateful for the resources and platform provided by the publisher, which have enabled us to share our findings with a wider audience. We appreciate the efforts of the editorial team in reviewing and editing our work, and we are thankful for the opportunity to contribute to the field of research through this publication.

Funding Information

The authors have not received any financial support or funding to report.

Author's Contributions

Nikhitha Pai: Participated in all experiments, coordinated the data-analysis and contributed to the writing of the manuscript.

Ilango Velchamy: Coordinated for research study and plan.

Nithya B Ramesh: Contributed for preparing the results and organization of the manuscript.

Availability of Data and Materials

Datasets are available on NSE and BSE sites. However, will be provided upon request.

Ethics

The paper reflects the authors' own research and analysis in a truthful and complete manner. The paper properly credits the meaningful contributions of co-authors. The results are appropriately placed in the context of prior and existing research.

References

Ahyar, L. F., Suyanto, S., & Arifianto, A. (2020, August). Firefly algorithm-based hyperparameters setting of DRNN for Weather Prediction. In *2020 International Conference on Data Science and its Applications (ICoDSA)* (pp. 1-5). IEEE. https://doi.org/10.1109/ICoDSA50139.2020.9212921

Bouktif, S., Fiaz, A., Ouni, A., & Serhani, M. A. (2018). Optimal deep learning LSTM model for electric load forecasting using feature selection and genetic algorithm: Comparison with machine learning approaches. *Energies*, *11*(7), 1636. https://doi.org/10.3390/en11071636

Brownlee, J. (2017). *Long short-term memory networks with python: Develop Sequence Prediction Models with Deep Learning*. Machine Learning Mastery. https://machinelearningmastery.com/ LSTMs with Python

Chou, J. S., & Nguyen, T. K. (2018). Forward forecast of stock price using sliding-window metaheuristic-optimized machine-learning regression. *IEEE Transactions on Industrial Informatics*, *14*(7), 3132-3142. https://doi.org/10.1109/TII.2018.2794389

Eiben, A. E., & Smith, J. E. (2015). *Introduction to evolutionary computing*. Springer-Verlag Berlin Heidelberg. https://doi.org/10.1007/978-3-662-44874-8

Gao, Y., Wang, R., & Zhou, E. (2021). Stock prediction based on optimized LSTM and GRU models. *Scientific Programming*, *2021*, 1-8. https://doi.org/10.1155/2021/4055281

Ghosh, A., Bose, S., Maji, G., Debnath, N., & Sen, S. (2019, September). Stock price prediction using LSTM on Indian share market. In *Proceedings of 32nd International Conference on* (Vol. *63*, pp. 101-110). https://doi.org/10.29007/qgcz

Hiransha, M., Gopalakrishnan, E. A., Menon, V. K., & Soman, K. P. (2018). NSE stock market prediction using deep-learning models. *Procedia Computer Science*, *132*, 1351-1362. https://doi.org/10.1016/j.procs.2018.05.050

Kaipa, K. N., & Ghose, D. (2017). *Glowworm swarm optimization: Theory, algorithms, and applications* (Vol. *698*). Springer. ISBN: 3319515942

Kaipa, K. N., Ghose, D., Kaipa, K. N., & Ghose, D. (2017). Glowworm swarm optimization: Algorithm development. *Glowworm Swarm Optimization: Theory, Algorithms and Applications*, 21-56. https://doi.org/10.1007/978-3-319-51595-3_2

Kedia, V., Khalid, Z., Goswami, S., Sharma, N., & Suryawanshi, K. (2018, August). Portfolio generation for Indian stock markets using unsupervised machine learning. In *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, (pp. 1-5). IEEE. https://doi.org/10.1109/ICCUBEA.2018.8697771

Krishnanand, K. N., & Ghose, D. (2005, June). Detection of multiple source locations using a glowworm metaphor with applications to collective robotics. In *Proceedings 2005 IEEE Swarm Intelligence Symposium, 2005. SIS 2005*, (pp. 84-91). IEEE. https://doi.org/10.1109/SIS.2005.1501606

Krishnanand, K. N., & Ghose, D. (2009). Glowworm swarm optimization for simultaneous capture of multiple local optima of multimodal functions. *Swarm Intelligence*, *3*, 87-124. https://doi.org/10.1007/s11721-008-0021-5

Roondiwala, M., Patel, H., & Varma, S. (2017). Predicting stock prices using LSTM. *International Journal of Science and Research (IJSR)*, *6*(4), 1754-1756.

Sagheer, A., & Kotb, M. (2019). Time series forecasting of petroleum production using deep LSTM recurrent networks. *Neurocomputing*, *323*, 203-213. https://doi.org/10.1016/j.neucom.2018.09.082

Seidy, E. E. (2016). A new particle swarm optimization based stock market prediction technique. *International Journal of Advanced Computer Science and Applications*, *7*(4). https://doi.org/10.14569/IJACSA.2016.070442

Shen, J., & Shafiq, M. O. (2020). Short-term stock market price trend prediction using a comprehensive deep learning system. *Journal of Big Data*, *7*(1), 1-33. https://doi.org/10.1186/s40537-020-00333-6

Stajkowski, S., Kumar, D., Samui, P., Bonakdari, H., & Gharabaghi, B. (2020). Genetic-algorithm-optimized sequential model for water temperature prediction. *Sustainability*, *12*(13), 5374. https://doi.org/10.3390/su12135374

Thakkar, A., & Chaudhari, K. (2021). A comprehensive survey on portfolio optimization, stock price and trend prediction using particle swarm optimization. *Archives of Computational Methods in Engineering*, *28*, 2133-2164. https://doi.org/10.1007/s11831-020-09448-8

Schaul, T., Bayer, J., Wierstra, D., Sun, Y., Felder, M., Sehnke, F., ... & Schmidhuber, J. (2010). PyBrain. *Journal of Machine Learning Research*, *11*(ARTICLE), 743-746. https://www.jmlr.org/papers/volume11/schaul10a/schaul10a.pdf

Wang, W., Li, W., Zhang, N., & Liu, K. (2020). Portfolio formation with preselection using deep learning from long-term financial data. *Expert Systems with Applications*, *143*, 113042. https://doi.org/10.1016/j.eswa.2019.113042

Xiao, D., & Su, J. (2022). Research on Stock Price Time Series Prediction Based on Deep Learning and Autoregressive Integrated Moving Average. *Scientific Programming*, *2022*. https://doi.org/10.1155/2022/4758698

Yadav, A., Jha, C. K., & Sharan, A. (2020). Optimizing LSTM for time series prediction in Indian stock market. *Procedia Computer Science*, *167*, 2091-2100. https://doi.org/10.1016/j.procs.2020.03.257

Yao, S., Luo, L., & Peng, H. (2018, August). High-frequency stock trend forecast using LSTM model. In *2018 13th International Conference on Computer Science & Education (ICCSE)*, (pp. 1-4). IEEE. https://doi.org/10.1109/ICCSE.2018.8468703

Yu, Y., Si, X., Hu, C., & Zhang, J. (2019). A review of recurrent neural networks: LSTM cells and network architectures. *Neural Computation*, *31*(7), 1235-1270. https://doi.org/10.1162/neco_a_01199

Yun, H., Lee, M., Kang, Y. S., & Seok, J. (2020). Portfolio management via two-stage deep learning with a joint cost. *Expert Systems with Applications*, *143*, 113041. https://doi.org/10.1016/j.eswa.2019.113041