Optimizing LSTM Based Network For Forecasting Stock Market

Ehsan Rokhsatyazdi
Department of Electrical, Computer and Software Engineering
Ontario Tech University, Oshawa, Canada
ehsan.rokhsatyazdi@ontariotechu.net

Shahryar Rahnamayan, SMIEEE
Department of Electrical, Computer and Software Engineering
Ontario Tech University, Oshawa, Canada
shahryar.rahnamayan@uoit.ca

Hossein Amirinia
Department of Electrical, Computer and Software Engineering
Ontario Tech University, Oshawa, Canada
hossein.amirinia@ontariotechu.net

Sakib Ahmed
Department of Electrical, Computer and Software Engineering
Ontario Tech University, Oshawa, Canada
sakib.ahmed@ontariotechu.net

***Abstract*— In this modern era, the financial market, more specifically, the stock markets all over the world, deal with an enormous amount of real-time data that facilitates the data analytics and prediction in the field of finance. The main objective of this paper is to propose a novel model of neural network based on Long-Short Term Memory (LSTM) and utilizing one of the most powerful evolutionary algorithms, namely the Differential Evolution (DE), to forecast the next day's stock price of a company. This study focuses on optimizing the ten network hyperparameters related to the detection of temporal patterns of a given dataset, namely, the size of the time window, batch size, the number of LSTM units in hidden layers, the number of hidden layers (LSTM and dense), dropout coefficient for each layer, and the network training optimization algorithm. To the best of our knowledge, this is the first time that all this set of parameters have been optimized simultaneously. Then, the LSTM has been optimized by DE to gain the lower root mean squared error (RMSE) for prediction. The proposed model achieved 8.092 RMSE as its objective value, which is better in comparison with the best statistical forecasting models such as NAIVE, ETS, and SARIMA, which are the-state-of-the-art methods in this filed. Moreover, for shortening the training time as the main source of computational expensiveness, the proposed method works with a lower number of epochs. By this way, DE tries to find a shallower and faster network even with higher accuracy, which is a remarkable approach.**

Keywords*— *Long Short-Term Memory (LSTM), Artificial Neural Network, Hyperparameter Optimization, Time Series Prediction, Statistical Forecasting Model, Differential Evolution (DE)

I. INTRODUCTION

Forecasting refers to the prediction of some upcoming events by analyzing the previous data. Forecasting can be basically three types: short-term, medium-term, and long-term forecasting. In this paper, stock price forecasting is discussed. Two main types of forecasting techniques for stock price prediction are fundamental analysis and technical analysis. Fundamental analysis is the most suited approach for long-term forecasting that analyzes sales, earnings, profits, and other economic factors of a company to determine the share value of it. In technical analysis, the future price of stocks is calculated by analyzing the historical price of stocks. Time series forecasting is a type of technical method. Generally, the time series forecasting analyzes the time series data, and they are classified into two models: linear and nonlinear models. A linear model cannot identify the patterns or dynamics existence in the data as a whole, whereas non-linear models such as deep learning methods that are capable of identifying hidden patterns and main dynamics in the data using a self-learning process [1].

Nowadays financial market creates an enormous amount of data that comprises transaction data, storage data, moving data, etc. With the advancement of computing technology and the invention of several advanced methods such as deep learning, it would be able to predict, calculate, and make decisions more efficiently regarding financial data [2]. In this paper, a unique stock price forecasting model using historical data is proposed.

Stock market prediction is always challenging because of its complex, chaotic, and dynamic nature [3]. Moreover, the volatility of the stock market is too large to set in a specific model that makes it more troublesome. Despite the challenges, several methods have been introduced to predict the stock market. Deep learning is one of them which has multiple hidden layers between the input and output layers. Recurrent neural network (RNN) is a well-known one in deep learning, which is mostly used for time-series analysis. This method can endure previous information with the help of feedback connections inside the network, and it also has non-linear prediction capabilities. RNN is advantageous in processing sequential data, which is not effectively possible using contemporary neural networks. That is why, in this paper, long short-term memory (LSTM) which is a type of RNN, is selected for sequence learning of financial time-series [2]. LSTM can recognize present and previous examples by providing different weights, and at the same time, it overlooks extra memory to anticipate the following output. That is why LSTM has better capability to deal with long-sequence input compared to other variants of RNN [3].

Despite being a useful tool in time series, LSTM networks have some limitations. LSTM networks belong to a high-level computational process to accomplish a solution for the targeted problem, but they are not ready to give explicit clarifications to their forecast results [2].

This study focuses on the optimization of network hyperparameters related to the detection of temporal patterns of a given dataset, such as the size of the time window and the number of LSTM units in hidden layers, The number of hidden layers (LSTM and dense), dropout coefficient of each layer (in Keras library) except for the output layer, network training optimization algorithm, for instance, Adam, Nadam, GSD and so on. DE is used to determine the best hyperparameters to be considered in a model, and simultaneously investigate the optimal number of hidden layers of the LSTM network. Appropriate size detection for the time window that can contain the context of the dataset is a crucial task when designing the LSTM network. It is essentially important to know that significant signals may be missed for small-time window size, whereas if the size of the time window is too large, unsuitable information may perform as noise.

This paper had the following contributions. Proposing a deep neural network with LSTM and dense layers for stock market prediction. Moreover, optimizing network hyperparameters such as batch size, window size, number of layers by the DE algorithm. DE is customized for this problem by generating integer numbers for some genes and float numbers for other genes of the chromosome. The tackled optimization problem in this paper is expensive in terms of computation time and mixed-type variables.

The remainder of this paper is organized as follows: a literature review of previous work on LSTM is presented in Section 2, while Section 3 illustrates the LSTM model and utilized a global optimization algorithm, i.e. DE, that have been used in stock market prediction. In Section 4, the details of the conducted experiments in aggregating the DE and LSTM network are presented. As a case study, the application of the proposed algorithm in the Tesla stock price together with the experimental analysis is demonstrated in Section 5, followed by the conclusions and future work.

II. Background Review

Stock market analysis is naturally challenging because of its complex, dynamic, and uncertain nature. Researchers have studied the stock market prediction for decades [2]. There are two known hypotheses regarding stock market prediction: the Efficient-Market hypothesis and the Random-walk hypothesis. The efficient-market hypothesis demonstrates that the present price of an asset depends on all historical information accessible for it [4]. Random-walk hypothesis defines that future prices will just rely upon future data irrespective of the present price [5].

Many studies have been conducted to predict future financial market analysis based on statistical and machine learning methodologies. Statistical analysis is popular to make predictions about the financial market based on historical stock data, such as the autoregressive integrated moving average model (ARIMA), the auto-regressive conditional heteroskedasticity (ARCH) model, and the generalized autoregressive conditional heteroscedasticity (GARCH) model [6], [7], [8]. The statistical method for predictability analysis needs more historical data for future forecasts.

As a nonlinear and non-parametric dynamic system, the stock market requires progressively adaptable techniques that can learn complex dimensionality for predictive analysis [9]. Machine learning could be an option to fulfill these requirements as it can figure out nonlinear relationships among data without having previous knowledge of the input data [10]. Among the many machine learning techniques that have been used until now, Artificial Neural Network (ANN) and Support Vector Machine (SVM) are selected because of their ability to examine the noisy behavior of information without making any statistical confinements while predicting financial time-series [11].

Hyejung et al. [2] proposed a hybrid methodology of the Genetic Algorithm (GA) and LSTM to predict next-day stock price by training with historical data. In this study, GA is used to figure out the optimal value of time- window size and the number of LSTM units in the LSTM network. In another study on stock price forecasting, researchers have used a Back Propagation (BP) in the first place. Later on, a new model based on LSTM was introduced to get better convergence and training efficiency [12]. An improved LSTM, aggregated with particle swarm optimization (PSO) and gradient descent (GD) was studied in [13]. In this study, the LSTM network is first trained using the GD approach for a few iterations, and then GD obtained model parameters, i.e., the weights and biases of the LSTM network, are passed to the PSO based optimization algorithm. PSO optimize the model parameters of the LSTM and send it to GD again [13]. Zhou et al. (2019) have recently proposed an approach where hyperparameters are optimized using Sequence Model-Based Optimization (SMBO) [14].

In [15], researchers have used Ant Colony Optimization (ACO) technique to improvising long short-term memory (LSTM) by refining their cell structure in an examination to analyze them as an answer for anticipating airplane motor vibration. Multi-objective Evolutionary Optimization methodology was used in another study where training and topology of RNN are optimized simultaneously following three concepts: (1) preservation of local optima Solutions, (2) elite preserve strategy, and (3) self-adaptive mutation probability setting [16]. A combined methodology of SVM and GA was proposed to forecast entry and exit points of Forex Market for an automatic investment system. In this study, the researchers used the SVM to classify, and GA is used to optimize the investment strategies [17]. Chou et al. (2016) introduced a novel sliding window metaheuristic based prediction system that can recognize energy consumption patterns by predicting energy consumption one day in advance by exploring time-series data of a smart grid [18]. The difference between previous works and this work is that the proposed method is more comprehensive in terms of the variety of hyperparameters, which have been optimized, and DE optimization method itself.

III. Proposed Method

A. Long Short-Term Memory (LSTM) Network

LSTM network is constructed with several LSTM units and is basically as a structure of RNN. RNN is a deep learning network, but it is different from traditional feed-forward networks. RNN has internal feedback among neurons in which they can pass data to a previous or the same layer. In this way, data, instead of flowing in a single direction, flow to the previous layer as well. The consequence of this phenomenon is the existence of short-term memory along with long-term memory that has been achieved by training [3].

LSTM network consists of a memory cell and three gating units: an input, an output, and a forget gate [2].

LSTM was first introduced by S. Hochrieter and J. Schmidhuber in order to provide a solution to vanishing gradient problem, which is suffered by RNN while working with long data sequences [19]. In this process, the gates keep the error flow constant that takes weights adjustments into consideration [3].

The input gate controls the amount of information that will flow from the inputs of the cell, and then the forget gate regulates how much information will flow from the cell-memory. Lastly, the output gate decides the amount of information will flow out of the cell. In this way, the network can get the idea of target value as well as the knowledge of tuning the controls to reach the target value [20].

B. *Differential Evolution (DE)*

Differential Evolution is a metaheuristic and stochastic optimization algorithm inspired by the process of natural evolution that optimizes an objective to a given standard by an iterative effort towards the improvement of a candidate solution. In many cases, it outperforms the traditional evolutionary algorithm. In this process, a combination of parent individuals and other individuals of the same population create a candidate solution. The parent could be replaced by a candidate solution if it has better fitness than the parent [21], [22].

DE is a simple, efficient, easy to implement method. DE operates with two populations. One is the old generation, and the other one is the new generation. The population comprises dimension D vectors, which are equal to the number of design parameters. The process of DE consists of three operating factors: mutation (in (1)), crossover (in (2)), and selection (in (3)). X_r is a solution vector in the current population, and V is mutant vectors. F stands as mutation factor, and $f(.)$ is a function that should be optimized, and in this paper, it is the network accuracy. U is next-generation vector after crossover, and $X_{i,G+1}$ is a solution vector in population for the next generation which is selected based on fitness function. CR is crossover probability [23].

$$V_{i,G+1} = X_{r1} + F \times (X_{r2} - X_{r3}) \tag{1}$$

$$U_{j,i,G+1} = \begin{cases} V_{j,i,G+1} \; if \;\; rand_{j,i} \leq CR \;\; or \;\; j = I_{rand} \\ X_{j,i,G+1} \; if \;\; rand_{j,i} > CR \;\; or \;\; j \neq I_{rand} \end{cases} \tag{2}$$

$$X_{i,G+1} = \begin{cases} U_{i,G+1} \; if \; f(U_{i,G+1}) < f(X_{i,G}) \\ X_{i,G} \;\; otherwise \end{cases} \tag{3}$$

In DE, the individuals of a population are regarded as target vectors. The mutation operation then creates a mutant vector for each target vector. On the other hand, the crossover operator produces a trial vector by integrating parameters of the mutant vector and target vector. This targeting vector can be replaced by a trial vector in the next generation if the trial vector achieves better fitness value than the target vector [23].

IV. Conducted Experiment

For this experiment, DE is utilized to find the best hyperparameters on the LSTM network for stock market data (Utilized Tesla stock market data from 01/04/2016 to 12/02/2019 consists of 986 data points). Figure 1 shows how optimizer interacts with the LSTM network to tune the hyperparameter. For implementation, Keras library in Python programming language has been used.

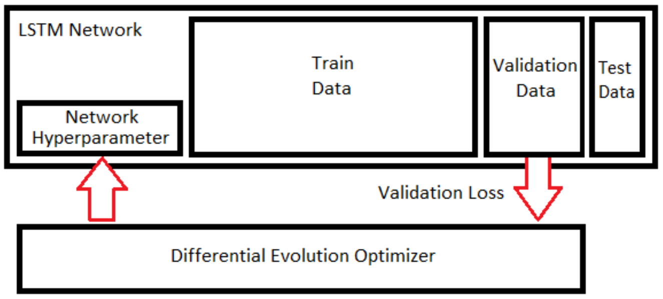

Fig. 1. Coupling of optimizer and LSTM to optimize network hyperparameter based on validation loss.

TABLE I. HYPERPARAMETRS AS DECISION VARIABLES

Gene name	Hyperparameter	Box-constraints
G1	Batch size	$10 \leq G1 \leq 150$
G2	Window size	$1 \leq G2 \leq 100$
G3	Layer combination (Fig. 2)	$1 \leq G3 \leq 4$
G4	Nodes of layer 1	$1 \leq G4 \leq 80$
G5	Nodes of layer 2	$1 \leq G5 \leq 50$
G6	Nodes of layer 3	$1 \leq G6 \leq 50$
G7	Drop out for Layer 1	$0 \leq G7 \leq 0.5$ (0, 0.1, … ,0.5)
G8	Drop out for Layer 2	$0 \leq G8 \leq 0.5$ (0, 0.1, … ,0.5)
G9	Drop out for Layer 3	$0 \leq G9 \leq 0.5$ (0, 0.1, … ,0.5)
G10	Network training optimization algorithm (1 to 6 are SGD, Adagrad, Adadelta, Adam, Nadam, Adamx respectively)	$1 \leq G10 \leq 6$

Data has been split into three parts. The first one is training data which has 665 data points, the second one is validation data (221 data points), and the last part is testing data which has 100 data points, and has been used as unseen data for network final accuracy testing.

Since the LSTM hyperparameters are integers (except for the dropout values), the DE produces integer numbers to generate the population. Ten genes are employed to make a chromosome, which is shown in Table 1. In none of the previous research works, all this set of parameters have been optimized simultaneously. The combination of these discrete hyperparameters is more than 5×10^{12} possibility, which makes the exhaustive search impossible. Moreover, because of the stochastic nature of the network, for each fitness call, the whole network should be trained five times (or more) and get the average of the error of these five runs to have a more robust and trustable result as the network error. This process makes the optimization procedure five times more expensive in terms of computational time.

G1 is the batch size on genotype since it has a great impact on network performance [24]. Windows size is the sliding window size. It looks back to the specific number of previous samples. The layer combination consists of two LSTM layers and two dense layers. In Figure 2, the phenotype of the G3 is shown. For the layers, four combinations are proposed for the network. For instance, the first combination is one LSTM and one dense layer. All these layer combinations are followed by a single node dense layer as the output of the network.

The layer nodes are the number of memory units on LSTM layers and the number of neurons on dense layers (except the dense output layer, which has just one node). When this number is equal to 4, the number of nodes of the third and fourth layers will be the same. The dropout is the dropout coefficient in each layer in order to boost generalization and avoid overfitting [25]. G10 is the type of network training optimizer algorithm since weights are updated by famous gradient descent such as Adam, Nadam, SGD, Adagrad, Adadelta, and Adamax.

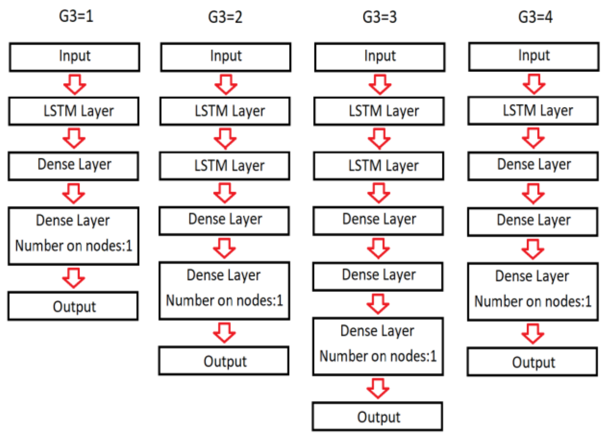

Fig. 2. Neural network layer architecture for G3=1 to G3=4.

DE is modified to generate integer numbers for genes G1 to G6 and G10. Among DE operands (in (1) to (3)), just mutation can generate non-integer numbers because of the mutation factor (F), which is a float number between 0 and 1. In this paper, the mutant vector rounded to an integer one to avoid generating non-integer numbers. For other genes (G7, G8 and G9), DE generates discrete real numbers (0.0, 0.1, …, 0.5). For this purpose, DE generates integer number between 0 and 5, then this number multiply by 0.1 to generate discrete float numbers.

Moreover, mutation operand can generate new vectors which can be out of the box-constraints. To address this issue, after mutation, if the generated amount of G1 to G10 are out of their box-constraint, the algorithm is forced to correct them by limiting them to the value of the bonds. Therefore, the value of genes is limited to their constraint. Now DE is ready, and it generates chromosomes (hyper-parameters) and passes them into the LSTM network. LSTM tries to predict the next day's price.

DE generates an initial population with 20 individuals and creates the next generation based on mutation and crossover operators of DE. To see the effect of the number of epochs, training process has been done with 70 epochs and ten epochs. The total number of generations is 100 when training epochs are 10. The total number of generations is 31 when training epochs are 70. In both cases, the training time is the same. For each DE fitness call, the average of the five runs is considered because of the stochastic nature of the neural network. As a result, the network is trained and validated 10,000 times with ten epochs, and then 1,600 times with 70 epochs. The configuration of the algorithm is shown in Table 2.

TABLE II. DE Algorithem Configuration

Parameters	**Value (10 epoch)**	**Value (70 epoch)**
Population size	20	20
Number of generations	100	31
Crossover probability (CR)	0.9	0.9
Mutation factor (F)	0.8	0.8
Number of independent training of LSTM per fitness call For each hyperparameter combination	5	5
Total Number of LSTM network training	10,000	1,600

The fitness function is the root mean squared error of the forecasting procedure that will be calculated for each individual combination of hyperparameters.

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}} \quad (4)$$

Root-mean-square-error (RMSE) is a standard error measurement metric for forecasting the quantitative data model (in (4)). In this equation, $\hat{y}_1, \hat{y}_2, ..., \hat{y}_n$ are predicted values, and $y_1, y_2, ..., y_n$ are observed values, and n is the number of observations.

For testing the models, Tesla stock historical data has been used. Data set consists of 986 days of closed price from Jan. 2016 to Dec. 2019.

V. Result Analysis

In this section, the proposed model has been compared with three famous statistical methods, namely, ETS, SARIMA, and NAIVE. These models define the baseline for comparing the performance of the proposed model with the most powerful statistical models. Makridakis et al. [26] have mentioned that these baseline models generally perform better than all of the machine learning and deep learning models (see Figure 3) for time series prediction on a wide variety of data set. Therefore, as far as the best models are the statistical ones, in this paper, these models have been chosen and optimized by grid search [27] for comparing purposes on specific stock market data set.

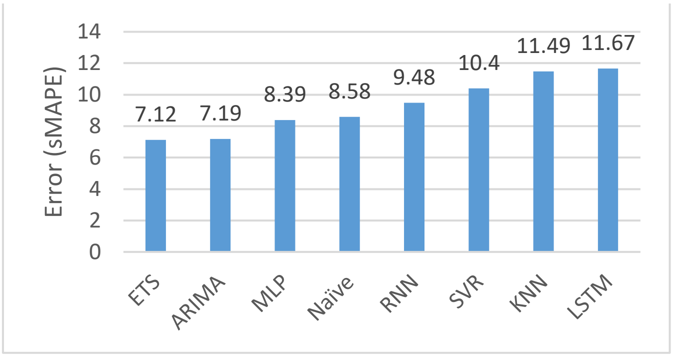

Fig. 3. Overal models' error for time series forecasting on a wide variety of data sets [26]. Statistical models perform better than other machine learning methods.

The statistical models have some parameters that should be set for better prediction performance. For finding these parameters, an exhaustive search method (grid search) has been used to find the best parameters. The best setting parameters for these models are shown in Tables 3, 4. Based on the expectations, because the stock market has not seasonality in general, the seasonal parameters are zero (or false).

TABLE III. ETS Best Settings

ETS Model Parameters	**Best Value based on an exhaustive search**
Trend	Multiplicative
Damped	True
Seasonal	None
Seasonal periods	None
Box-Cox Transform	False
Remove Bias	False

TABLE IV. SRIMA Best Setings

SARIMA Model Parameters	**Best Value based on an exhaustive search**
Trend Order	autoregression = 2
	difference = 0
	moving average = 2
Seasonal Order	autoregressive = 2
	difference = 0
	moving average = 0
	time steps for a single seasonal period = 0
Trend parameter	no trend

As has been mentioned above, DE is used to optimize the Hyperparameter of the network because it is almost impossible to find them by exhaustive search. All combinations of the Hyperparameters can not be tested one by one in a grid search. Moreover, for each of the combinations, because of the stochastic nature of neural networks, the model should be trained and tested a couple of times (5 times In the proposed method) to get the average performance of the network so the computational cost would be very high, but DE makes it possible to find optimal hyperparameters. DE performance plot is shown in Figure 4. In this figure, the validation error has been shown generation by generation. The training network is demanding in terms of time and computational power. Two different number of epochs has been selected (70 and 10) to see their effects on the results. Both algorithms are given almost the same budget by stopping the algorithm at the same time (263.2 hours) on the same system (Intel Core i5-6400 CPU with 16GB of RAM). For the training method with ten epochs, the number of generations was 100, and at the same time, the training method with 70 epochs could cover 31 generations. The empirical experiments showed an interesting result that when a low number of epochs have been set for training, the DE algorithm tries to find the shallower network even with the lower error.

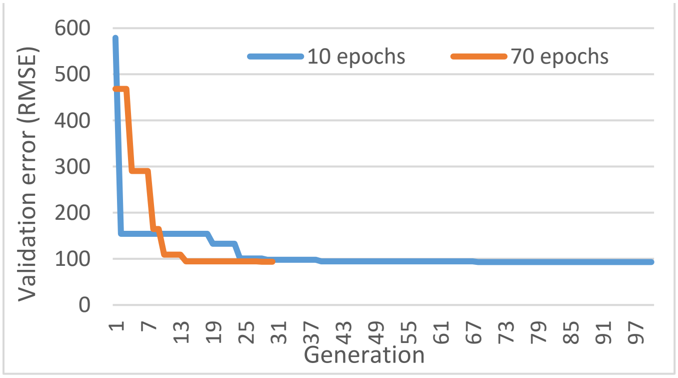

Fig. 4. DE performance plot for a different number of epochs. Both tests have an equal training time.

TABLE V. BEST CHROMOSOME

Best Chromosome (number of epochs = 70)									
G1	G2	G3	G4	G5	G6	G7	G8	G9	G10
40	8	4	50	48	38	0	0.5	0.3	4
Best Chromosome (number of epochs = 10)									
G1	G2	G3	G4	G5	G6	G7	G8	G9	G10
10	64	1	67	49	1	0	0.3	0.4	3

Based on these empirical results, proposed LSTM can be used to predict a very complex time series like the stock market. Fig. 5 illustrates the real data and predicted values on test data for the Tesla market index, which achieved by the proposed optimized LSTM network. After the optimization phase, for better training performance, the validation data concatenate to training data, and all of them have been used as a training set for better final model adaptation.

DE automatically performs exploration and then exploitation phase in the hyperparameter search space to find a better combination or even the best one. The best result after the optimization procedure on the LSTM model is less than other statistical methods (Fig. 6), and the related chromosomes are given in Table 5. It shows that when the DE algorithm forced to find the best result by a smaller number of epochs, it tries to find the optimal shallower network (3 layers when training has been done by ten epochs and four layers when the number of epochs is 70). G3 is related to the layer combination.

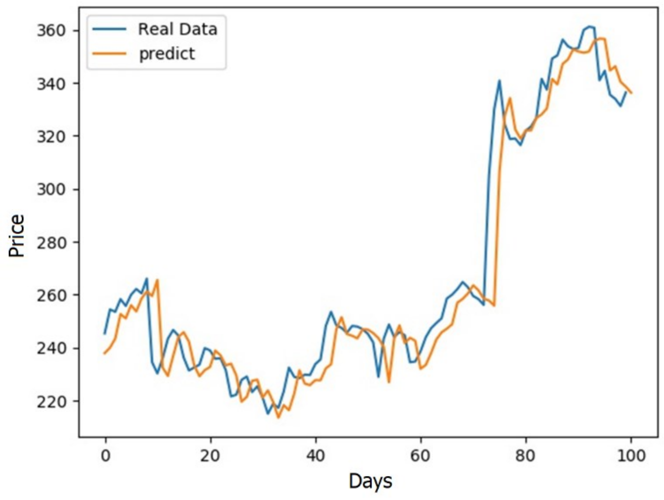

Fig. 5. Real and predicted values on test data.

The results of experiments show that the proposed model with ten epochs for training is more accurate with 8.092 RMSE, the proposed model with 70 epochs for training is the second-best, and SARIMA is the third-best, while those of ETS and NAIVE are 8.4 and 8.519 respectively. This result is noticeable because it shows that by using optimization methods, Machine learning methods can outperform statistical methods, which are in contrast with previous research results [26].

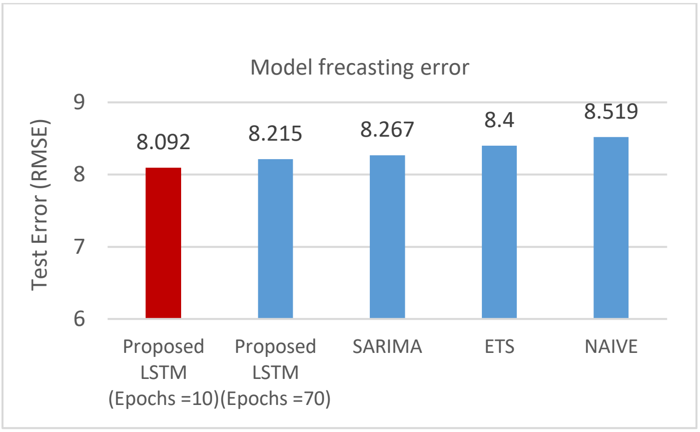

Fig. 6. Model error comparison: The proposed model with ten epochs for training during 100 generations outperformed other models, including proposed LSTM with 70 epochs during 31 generations.

VI. CONCLUSION AND FUTURE WORK

In the time series forecasting field, the stock market prediction is one of the most challenging applications, and the machine learning algorithm showed weak performance for these problems. This research focuses on finding the best hyperparameters of deep neural networks which provides a new idea for the use of differential evolution algorithms for optimizing LSTM based network to predict one step ahead of the target stock. Results demonstrate that an optimized LSTM network outperforms statistical methods such as SARIMA, ETS, and NAIVE. These statistical models are a the-state-of-the-art competitor for machine learning algorithms. Furthermore, it can find the shallower network with even better accuracy when it forced to find the result more quickly. This is very remarkable because optimization of the neural network is very demanding in terms of time and computational power.

For future work, because of the high computational cost of the problem, parallel computing will be included to enhance efficiency and response time. Also, hybridization of the different models such as CNN-LSTM, ARIMA-LSTM models would be beneficial to get a more accurate result. Moreover, Using different methods to enhance the DE algorithm, like using center-based population generation mechanisms and noisy optimization methods can be investigated accordingly.

REFERENCES

[1] Selvin, Sreelekshmy and e. al, "Stock price prediction using LSTM, RNN and CNN-sliding window model," in *International Conference on Advances in Computing, International Conference on Advances in Computing. IEEE*, 2017.

[2] Chung, Hyejung and K.-s. Shin, "Genetic algorithm-optimized long short-term memory network for stock market prediction," *Sustainability*, 2018.

[3] Nelson, D. MQ, A. C. Pereira and R. A. d. Oliveira, "Stock market's price movement prediction with LSTM neural networks," in *International Joint Conference on Neural Networks (IJCNN). IEEE*, 2017.

[4] Fama, B. G and M. E. F, "Efficient Capital Markets: A Review of Theory and Empirical Work," *The Journal of Finance,* 1970.

[5] L. A. W and A. C. MacKinlay, A non-random walk down Wall Street, Princeton University Press, 2002.

[6] J. N. K. Rao, G. E. P. Box and G. M. Jenkins, "Time Series Analysis: Forecasting and Control," *Econometrica,* vol. 40, 1972.

[7] R. F. Engle, "Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation," *Econometrica: Journal of the Econometric Society,* 1982.

[8] Dijk and P. H. F. D. Van, "Forecasting stock market volatility using (non-linear) Garch models," *Journal of Forecasting,* 1996.

[9] Atiya, Y. S and A.-M. F, "Introduction to financial forecasting," *Applied Intelligence,* 1996.

[10] Atsalakis, G. S. and K. P. Valavanis, "Surveying stock market forecasting techniques–Part II: Soft computing methods," *Expert Systems with Applications,* 2009.

[11] E. Tay and L. Cao, "Application of support vector machines in financial time series forecasting," *Omega,* 2001.

[12] Y. Wang, Y. Liu, M. Wang and R. Liu, "LSTM Model Optimization on Stock Price Forecasting," in *2018 17th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES), IEEE*, 2018.

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

[14] S. Zhou, L. Zhou, M. Mao, H.-M. Tai and Y. Wan, "An Optimized Heterogeneous Structure LSTM Network for Electricity Price Forecastin," *IEEE Access,* 2019.

[15] A. ElSaid, F. E. Jamiy, J. Higgins, B. Wild and T. Desell, "Using ant colony optimization to optimize long short-term memory recurrent neural networks," in *GECCO 18: Proceedings of the Genetic and Evolutionary Computation Conference*, 2018.

[16] H. Katagiri, I. Nishizaki, T. Hayashida and T. Kadoma, "Multiobjective evolutionary optimization of training and topology of recurrent neural networks for time-series prediction," *The Computer Journal,* vol. 55, 2011.

[17] B. J. d. Almeida, R. F. Neves and N. Horta, "Combining Support Vector Machine with Genetic Algorithms to optimize investments in Forex markets with high leverage," *Applied Soft Computing,* vol. 64, 2018.

[18] J.-S. Chou and N.-T. Ngo, "Time series analytics using sliding window metaheuristic optimization-based machine learning system for identifying building energy consumption patterns," *Applied Energy,* vol. 177, 2016.

[19] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation,* vol. 9, 1997.

[20] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink and J. Schmidhuber, "LSTM: A search space odyssey," *IEEE Transactions on Neural Networks and Learning Systems,* vol. 28, 2017.

[21] R. Storn and K. Price, "Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces," *Journal of Global Optimization volume,* vol. 11, 1997.

[22] T. Robič and B. Filipič, "DEMO: Differential Evolution for Multiobjective Optimization," in *International Conference on Evolutionary Multi-Criterion Optimization*, 2005.

[23] U. K. Rout, R. K. Sahu and S. Panda, "Design and analysis of differential evolution algorithm based automatic generation control for interconnected power system," *Ain Shams Engineering Journal,* vol. 4, 2013.

[24] E. Hoffer, T. Ben-Nun, I. Hubara, N. Giladi, T. Hoefler and D. Soudry, "Augment your batch: better training with larger batches," *ArXiv:1901.09335,* 2019.

[25] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research,* vol. 15, 2014.

[26] S. Makridakis, E. Spiliotis and V. Assimakopoulos, "Statistical and Machine Learning forecasting methods: Concerns and ways forward," *PLOS ONE,* vol. 13, 2018.

[27] Brownlee and Jason, Deep Learning for Time Series Forecasting, Machine Learning Mastery, 2019.