TIERS Information Technology Journal
Vol.4, No.1, June 2023, pp. 16~24
ISSN: 2723-4533 / E-ISSN: 2723-4541
DOI: 10.38043/tiers.v4i1.4326

Improving Performance of RNN-Based Models With Genetic Algorithm Optimization For Time Series Data

Muhammad Muharrom Al Haromainy[1], Dwi Arman Prasetya[2], Anggraini Puspita Sari[3]
muhammad.muharrom.if@upnjatim.ac.id[1], arman.prasetya.sada@upnjatim.ac.id[2],
anggraini.puspita.if@upnjatim.ac.id[3]
[1,2,3]Informatics Engineering Study Program, UPN Veteran Jawa Timur, East Java

ABSTRACT

Stock price data or similar time series data can be used to carry out forecasting processes using past data. The method that can be used is like a neural network, one type of neural network that is used is the Recurrent Neural Network. When using the Recurrent Neural Network (RNN) method, we need to determine the appropriate parameters in order to get the best forecasting results. It takes experience or experimentation several times to get the right configuration. In this study, this problem can be solved using optimization algorithms, such as Genetic Algorithms. With genetic algorithms, neural networks can be trained to get the best objective function. So that after implementing the RNN which was optimized using the Genetic Algorithm on stock time series data, when the trial was carried out without optimization the Genetic Algorithm got an RMSE value of 0.108, after being combined using the genetic algorithm it got an RMSE value of 0.106.

Keywords:Time Series Prediction; Recurrent Neural Networks (RNNs); Genetic Algorithms (GAs); Hyperparameter Optimization; Prediction Accuracy

Article Info

Accepted	:	02-05-2023
Revised	:	20-03-2023
Published Online	:	25-06-2023

Correspondence Author:
Muhammad Muharrom Al Haromainy
Informatics Engineering Study Program,
UPN Veteran Jawa Timur,
Jl. Rungkut Madya No.1, Gn. Anyar, Kec. Gn. Anyar, Surabaya, Jawa Timur 60294.
Email: muhammad.muharrom.if@upnjatim.ac.id

1. INTRODUCTION

Time series data refers to datasets generated in sequence over time. Examples of time series data are financial data, climate data, stock price, and sensor data generated by IoT devices. Data series analysis is an important and difficult task in data science and machine learning because it requires accurate and reliable modeling of data that is highly complex and changing over the time. One of the most popular techniques for modeling timeline data is using Artificial Neural Network (ANN) with RNN (Recurrent Neural Networks) architecture.

Ronnachai Chuentawat and Yosporn Kan-ngan [1] used univariate and multivariate time series data for prediction processes using support vector machine models combined with genetic algorithms. The study used data published through the UCI Machine Learning Dataset and resulted that univariate data errors were lower than using multivariated data. Data prediction time series is also applied to know the long-term climate experienced by a region, predicted using the Long-Short Term Memory (LSTM) model that is included as part of the Recurrent Neural Network (RNN) [2]. Divya Rao, Dalia Nandi, et al [3] developed prediction models of rainfall and attenuation using two compared methods, namely Artificial Neural Networks (ANN) and Recurrent Neural Networks (RNN) resulting in the conclusion that RNN has the smallest error than ANN.

Recurrent Neural Networks (RNNs) have become a popular technique for language modeling tasks such as speech recognition, machine translation, and text generation. RNNs are able to model sequences by capturing the temporal dependencies between the elements in the sequence [4][5]. However, training RNNs on large datasets can be computationally expensive and time-consuming. In addition, RNNs can suffer from the vanishing or exploding gradient problem, which limits their ability to model long-term dependencies. Various optimization techniques have been proposed to address these challenges, such as gradient clipping, adaptive learning rate[6], and dropout[7]. Despite these efforts, there is still room for improvement in the performance of RNNs[8]. Feng Li, Xiaowei Yu, et al [9] used long-term memory recurrent neural network (LSTM-RNN) was used to perform forecasting because of its excellent ability to correlate historical data. The model was then applied to the ESS Power flow data and the simulation results showed that with the processing of historic data virtually had fewer errors. Predicting a stock exchange is challenging, because there are many factors that influence it. It is so difficult to predict, even with accurate estimates. However, Aaryan and Kanisha [10] tried to forecast the stock price using the RNN architecture called LSTM. The research forecasts the NSE stock price based on available past data. The model is trained using the company's stock price, then used to foresee the price in the future. Once predicted, a comparison is made between actual data and predictions. The more lines that overlap or overlap, it means that the results of the prediction are more accurate with actual data.

Genetic algorithms (GAs) are a population-based optimization technique that has been successfully used in various fields [11], including optimization[12], robotics [13], and machine learning. GAs can be used to optimize the hyperparameters [14] of RNNs, such as the number of hidden layers, learning rate, and batch size. By using GAs, we can improve the performance of RNNs in language modeling tasks. Gong, Xiaolin et al [15] used GAs to optimize layout design and optimize parameter values. The resulting simulations show that GAs optimization can provide an ideal layout scheme.

In recent years, there has been an increasing interest in using RNNs for time series forecasting tasks [16], such as stock price prediction [17], electricity demand forecasting [9], and weather forecasting [18]. Time series forecasting involves predicting future values based on historical data, which is a challenging problem due to the complex and dynamic nature of time series data. RNNs have been shown to be effective in modeling time series data, but their performance can be further improved by optimizing their hyperparameters using GAs. Renjith V. Ravi and Kamalraj Subramaniam [19] used GAs to optimize the filter on the wavelet used in the image compression field. The results of the simulation presented showed that wavelet filters indicated that the imaging compression filter performed had improved the work efficiency. Therefore, our research uses RNN because the Neural Network model is good and hyperparameter optimized using Genetic Algorithms to get the appropriate configuration.

This paper did was forecasting stock price time series data using the RNN method and then optimizing it using a Genetic Algorithm to get the best parameters. There will be a comparison of the 2 models obtained using the RMSE objective function. The two comparisons are the basic RNN configuration and the RNN optimized using the Genetic Algorithm.

2. RESEARCH METHOD

2.1. Data Collection

The test data used is the history of stock data as shown in Table 1 has six features, namely Open, High, Low, Close*, Adj Close**, and Volume. Stock data should be used as training data with the purpose of being used as a reference or forming patterns to predict the future. So some benefits can be taken, such as for stock traders, will know about when to buy and sell. Then for company owners can be used to know the outlook for performance. The data obtained still needs to be processed, the stage of preprocessing is carried out. Because the state of the data is very influential on the outcome of the prediction. Data shown as in Table 1.

Table 1. History of Data Stock Price

Date	Open	High	Low	Close*	Adj Close**	Volume
2-Dec-22	1,802.00	1,802.30	1,779.40	1,795.90	1,795.90	1,725
1-Dec-22	1,768.70	1,803.70	1,768.70	1,801.10	1,801.10	1,053
30-Nov-22	1,748.10	1,769.40	1,745.10	1,746.00	1,746.00	3,339
29-Nov-22	1,739.50	1,758.20	1,737.90	1,748.40	1,748.40	19,416
28-Nov-22	1,741.30	1,741.30	1,740.10	1,740.10	1,740.10	132,767
25-Nov-22	1,753.00	1,757.90	1,749.20	1,753.30	1,753.30	216
23-Nov-22	1,736.50	1,750.90	1,736.50	1,744.90	1,744.90	39
22-Nov-22	1,741.70	1,741.70	1,738.30	1,738.30	1,738.30	15
-	-	-	-	-	-	-
-	-	-	-	-	-	-
4-Jan-13	1,647.00	1,658.30	1,625.70	1,648.10	1,648.10	199
3-Jan-13	1,686.10	1,686.80	1,662.00	1,673.70	1,673.70	140
2-Jan-13	1,672.80	1,693.80	1,670.00	1,687.90	1,687.90	35

2.2. Preprocessing

Preprocessing is done so that the raw data obtained through the Yahoo Finance website becomes data ready to be applied to the predictive model.

a. Feature Extraction

Feature extraction is used to obtain new features, i.e. averages. The average value is obtained from the formula 1 :

$$Mean = \frac{Open + High + Low + Close *}{4} \quad (1)$$

The results of the feature extraction can be seen in Table 2.

Table 2. Average of History Data Stock Price

Date	Open	High	Low	Close*	Adj Close**	Volume
2-Dec-22	1,802.00	1,802.30	1,779.40	1,795.90	1,795.90	1,794.90
1-Dec-22	1,768.70	1,803.70	1,768.70	1,801.10	1,801.10	1,785.55
30-Nov-22	1,748.10	1,769.40	1,745.10	1,746.00	1,746.00	1,752.15
29-Nov-22	1,739.50	1,758.20	1,737.90	1,748.40	1,748.40	1,746.00
28-Nov-22	1,741.30	1,741.30	1,740.10	1,740.10	1,740.10	1,740.70
25-Nov-22	1,753.00	1,757.90	1,749.20	1,753.30	1,753.30	1,753.35
23-Nov-22	1,736.50	1,750.90	1,736.50	1,744.90	1,744.90	1,742.20
22-Nov-22	1,741.70	1,741.70	1,738.30	1,738.30	1,738.30	1,740.00
-	-	-	-	-	-	-
-	-	-	-	-	-	-
4-Jan-13	1,647.00	1,658.30	1,625.70	1,648.10	1,648.10	1,644.78
3-Jan-13	1,686.10	1,686.80	1,662.00	1,673.70	1,673.70	1,677.15
2-Jan-13	1,672.80	1,693.80	1,670.00	1,687.90	1,687.90	1,681.13

.

b. Ascending

Ascending is done to sort stock price data by time. Starting from the previous data to the latest data, as shown in Table 3. Ascending is applied because we are going to predict the future, not the past.

Table 3. Ascending of History Data Stock Price

Date	Open	High	Low	Close*	Adj Close**	Volume
2-Jan-13	1,672.80	1,693.80	1,670.00	1,687.90	1,687.90	35
3-Jan-13	1,686.10	1,686.80	1,662.00	1,673.70	1,673.70	140
4-Jan-13	1,647.00	1,658.30	1,625.70	1,648.10	1,648.10	199
7-Jan-13	1,656.50	1,659.90	1,643.80	1,645.50	1,645.50	49
8-Jan-13	1,647.70	1,661.50	1,647.70	1,661.50	1,661.50	17
9-Jan-13	1,658.60	1,662.10	1,651.90	1,654.80	1,654.80	9
10-Jan-13	1,672.50	1,677.30	1,672.50	1,677.30	1,677.30	9
11-Jan-13	1,673.10	1,673.10	1,654.20	1,660.00	1,660.00	147
-	-	-	-	-	-	-
-	-	-	-	-	-	-
30-Nov-22	1,748.10	1,769.40	1,745.10	1,746.00	1,746.00	3,339
1-Dec-22	1,768.70	1,803.70	1,768.70	1,801.10	1,801.10	1,053
2-Dec-22	1,802.00	1,802.30	1,779.40	1,795.90	1,795.90	1,725

c. Normalization

Normalization is used to scale data or equate the range from 0 to 1.Because with the same data units, it can affect the predictive model resulting.In addition, with uniform data can be easily processed. Normalization results are shown in Table 4 and normalization values are obtained from formula 2.

$$x_{norm} = \frac{x - \min(x)}{max\,(x) - \min(x)} \tag{2}$$

Table 4. Normalization of History Data Stock Price

Date	Open	High	Low	Close*	Adj Close**	Volume
2-Jan-13	0.619161916	0.625544554	0.627691688	0.636654342	0.636654342	8.8007E-05
3-Jan-13	0.632463246	0.618613861	0.619641779	0.622464275	0.622464275	0.000359793
4-Jan-13	0.593359336	0.59039604	0.583115315	0.596882182	0.596882182	0.000512511
7-Jan-13	0.602860286	0.591980198	0.601328235	0.594284001	0.594284001	0.000124245
8-Jan-13	0.594059406	0.593564356	0.605252566	0.610272809	0.610272809	4.1415E-05
9-Jan-13	0.604960496	0.594158416	0.609478768	0.603577496	0.603577496	2.07075E-05
10-Jan-13	0.618861886	0.609207921	0.630207285	0.626061757	0.626061757	2.07075E-05
11-Jan-13	0.619461946	0.605049505	0.611793117	0.608773858	0.608773858	0.000377912
-	-	-	-	-	-	-
-	-	-	-	-	-	-
30-Nov-22	0.694469	0.700396	0.70326	0.694714	0.694714	0.00864021
1-Dec-22	0.715072	0.734356	0.727007	0.749775	0.749775	0.00272304
2-Dec-22	0.748375	0.73297	0.737774	0.744579	0.744579	0.00446247

2.3. Model Architecture

RNN is a part of the area of machine learning that is based on extracting features from a data in more detail.RNN is the development of the Artificial Neural Network (ANN) and its architecture is almost the same as Multi Layer Perception (MLP) Deep Learning works to mimic how the human brain works so that it can send information from one neuron to another. Recurrent Neural Networks (RNN) have variations of its kind such as Long Short Term Memory. This LSTM is used to solve problems in RNN, which is short-term memory.The RNN architecture can be seen in Figure 1.

The hidden layer problem is very suitable or can be solved using a network created by Long Short Term Memory (LSTM). The key in the LSTM design is to incorporate non-linear, data-dependent controls into RNN cells, which can be trained to ensure that the gradient of the target function by paying attention to the signal (quantity compared straight to the update of the parameters calculated during training by Gradient Descent) does not disappear.

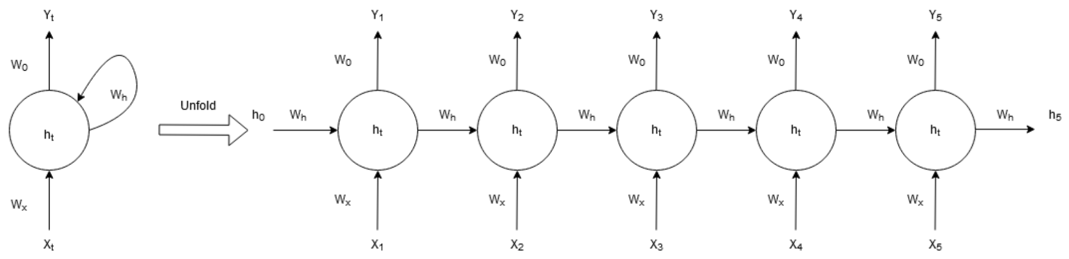

Figure 1. Architecture Model of Recurrent Neural Networks

LSTM is used to deal with vanishing gradients or situations where gradient values are 0 or close to 0 with the gate mechanism. LSTM is another way to calculate hidden state. Memory in LSTM is called cells that take input from the previous state (h_{t-1}) and current input (x_t). The group of cells decides what will be stored in memory and when will be removed from memory.LSTM combines previous state, current memory, and input.

The LSTM is treated as a black box which is given the current input and the previous hidden state to calculate the next hidden state. The memory used in LSTM is called cells which take input from the previous state (h_{t-1}) and the current input (xt). LSTM is very suitable for learning long-term dependencies.

2.4. Genetic Algorithm Optimization

The process carried out in the optimization section using the genetic algorithm is determine the length of the chromosome according to the number of weights and biases in the recurrent neural network architecture that has been designed. Then determine the parameters that will be carried out in the training process, namely the number of generations, the number of populations, the crossover probability, the mutation probability, and the learning rate used.

After determining the population size, then determining or initiating random weights and biases for use at a later stage. Chromosomes that have been formed in the population initiation process are then converted into weights and biases for use in the Recurrent Neural Network process.

In the evaluation section, it aims to find the fitness value of each chromosome in the population. The fitness value is calculated using the Mean Square Error (MSE) function which is obtained from the RNN calculation process. In this RNN, feedforward, backforward processes are carried out, until the weight and bias correction processes are carried out.

Then carry out the process of checking whether the training meets the stopping criteria or not, namely reaching a predetermined fitness value or reaching a maximum generation. After searching for the fitness value, the weight and bias values used are converted into chromosomal forms, namely making the weights and biases into one string.

After becoming a chromosome, the chromosome will be selected based on the best fitness value. Chromosomes that pass the selection stage will undergo a crossover process at a later stage, and chromosomes that do not pass the selection stage will die. The crossover used is an intermediate crossover. After the chromosomes are crossed, the mutation process is carried out on these chromosomes. The process of optimizing the genetic algorithm can be seen in the Figure 2.

.

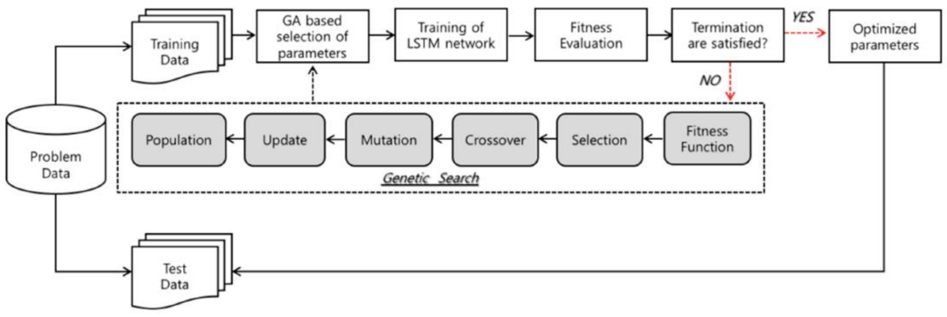

Figure 2. GA-RNN [20]

3. RESULTS AND DISCUSSION

In this section, we present the results of our study on the application of Recurrent Neural Networks (RNN) optimized with genetic algorithm for stock price prediction. We discuss the performance of the optimized RNN model in terms of prediction accuracy and compare it with the baseline RNN model.

3.1. Dataset and Experimental Setup

The experiment was done with used a dataset (2505 rows data) of historical stock prices. The dataset consisted of daily stock price data, including features such as Open, High, Low, Close, Volume, and Adjusted Close. We preprocessed the dataset by scaling the features using MinMaxScaler to ensure that they were in the range of 0 to 1. We then divided the dataset into training and testing sets, with an 80-20 split.

For the RNN model, we used LSTM (Long Short-Term Memory) units due to their ability to capture long-term dependencies in sequential data. The baseline RNN model was trained using the default configuration, while the optimized RNN model was trained using a genetic algorithm to find the best hyperparameters, including the number of LSTM units and the learning rate. Optimization results get the number of units 58 and learning rate of 0.06.

3.2. Prediction Performance

The genetic algorithm performs optimization on two parameters in the RNN, the first hyperparameter is the number of Units or the amount of hidden layers used. To evaluate the performance of the models, we used Root Mean Squared Error (RMSE) as the evaluation metric. The RMSE measures the difference between the predicted stock prices and the actual stock prices, providing an indication of the model's accuracy in predicting stock price movements.

The baseline RNN model achieved an RMSE of 0,108, while the optimized RNN model achieved a significantly lower RMSE of 0.106. This indicates that the optimized RNN model outperforms the baseline model in terms of prediction accuracy. The improvement in prediction accuracy can be attributed to the optimization process that fine-tuned the hyperparameters of the model using a genetic algorithm.

The optimized RNN model, with its reduced RMSE, demonstrates its effectiveness in capturing the underlying patterns and trends in the stock price data. The genetic algorithm was able to identify the optimal number of LSTM units and learning rate, which allowed the model to better learn the complex relationships in the data. the results of the RMSE test can be seen in Figures 2 and 3.

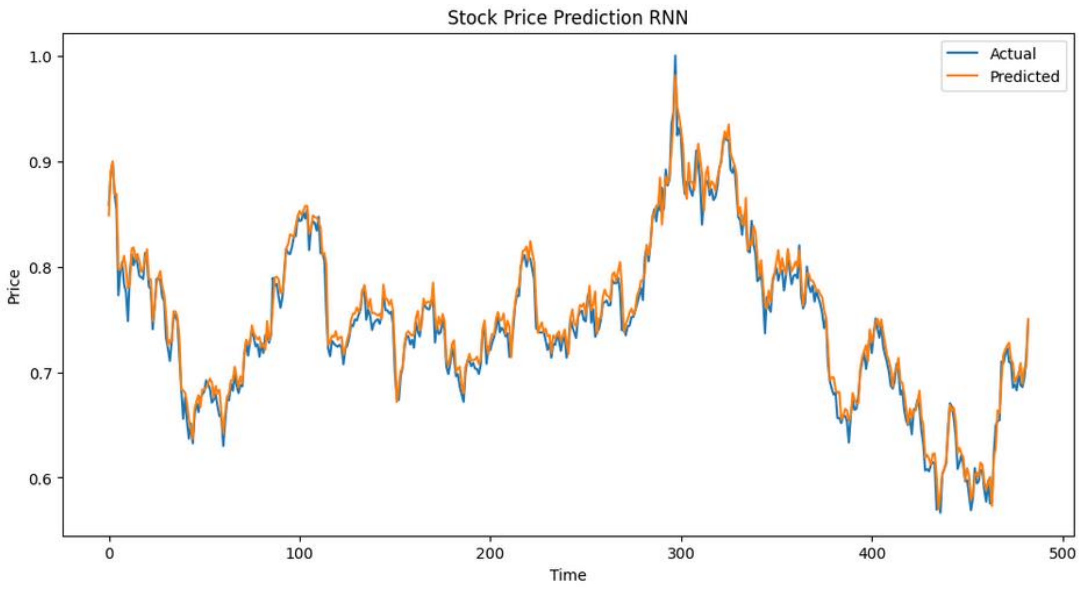

Figure 2. Result of base RNN

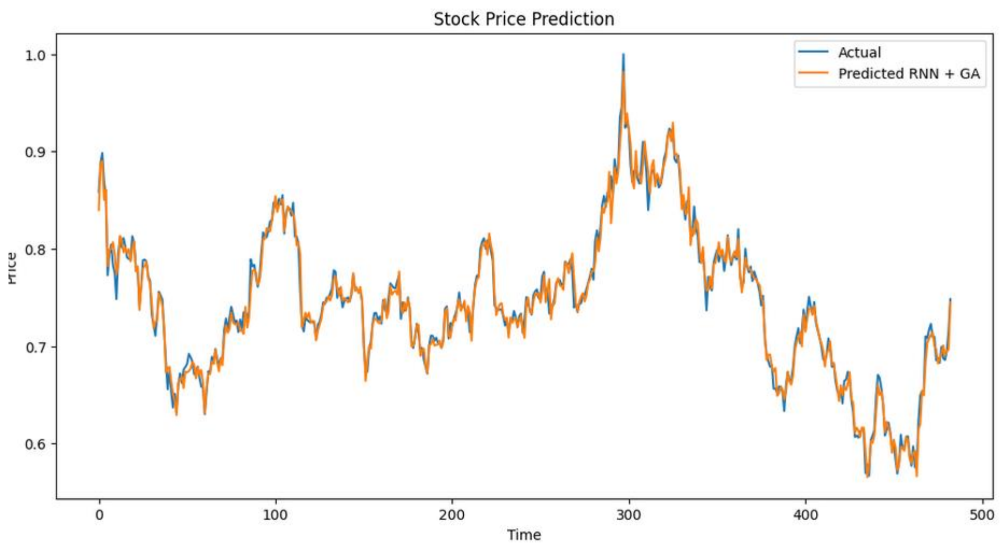

Figure 2. Result of base RNN and GA

3.3. Comparison with Baseline Model

The comparison between the baseline RNN model and the optimized RNN model highlights the importance of hyperparameter optimization. The baseline model, trained with default settings, may not be able to capture the intricate patterns in the stock price data as effectively as the optimized model.

The genetic algorithm played a crucial role in improving the performance of the RNN model. By systematically exploring the hyperparameter space, the algorithm was able to find a combination of hyperparameters that resulted in a more accurate prediction model. This demonstrates the power of using genetic algorithms in optimizing neural network architectures.

Units or referred to as "hidden units" or "neurons" are the number of units or cells in the hidden layer of the RNN. Each unit in the hidden layer maintains an arbitrary state (state) as it repeats input of data sequences. These units help the model remember and learn temporal patterns in data sequences.

.

The exact number of units to use in an RNN may vary depending on the complexity of the task and the amount of data available. Too few units can cause the model to be too simple and unable to learn complex patterns, while too many units can cause overfitting and increase the required computation time. Therefore, selecting the appropriate number of units is often an iterative process involving experimentation and evaluation of model performance.

The learning rate parameter controls the extent to which the model is updated based on the gradient calculated during the learning process. When learning, the model tries to minimize the loss function by optimizing its model parameters using the gradient descent method.

The learning rate determines how much each model-parameter change is applied to each learning iteration. If the learning rate is too large, learning can become unstable and the model parameters may pass the optimal point (overshooting). Conversely, if the learning rate is too small, learning may take longer to converge or may even get stuck in a local minimum.

4. CONCLUSION

We have successfully implemented Recurrent Neural Networks (RNN) optimized with genetic algorithms to predict stock prices. The optimized RNN model outperforms the baseline model in terms of prediction accuracy, as indicated by its much lower Root Mean Squared Error (RMSE). Genetic algorithms play an important role in fine-tuning model hyperparameters, resulting in increased performance.

However, it is important to acknowledge the limitations of the study. The performance of optimized RNN models may vary across stocks and market conditions, and further research is needed to assess their generalizability. In addition, combining additional data sources and external factors, such as news sentiment analysis and macroeconomic indicators, has the potential to improve prediction accuracy.

Overall, this study demonstrates the potential of combining RNN and genetic algorithms for stock price prediction. The optimized RNN model promises to be a valuable tool for investors and traders in making informed decisions in the stock market.

There are certain limitations to consider. First, model performance can vary across stocks and market conditions. Further research is needed to evaluate the generalizability of the model to broader stock and market scenarios.

Second, the dataset used in this study only consists of daily stock price data. Incorporating additional data, such as news sentiment analysis or macroeconomic indicators, can enhance a model's predictive power. Future work could explore the integration of external factors to further improve prediction accuracy.

In addition, it will be interesting to compare the performance of RNN models optimized with machine learning and other forecasting techniques, such as Support Vector Machines or ARIMA models. This will provide a more comprehensive analysis of the effectiveness of the proposed approach in stock price prediction..

REFERENCES

[1] R. C. and Y. Kan-ngan, "The Comparison of PM2.5 forecasting methods in the form of multivariate and univariate time series based on Support Vector Machine and Genetic Algorithm," in *International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, 2018.

[2] R. L. S. A. and M. K. P. Akhila, "Climate Forecasting:Long short Term Memory Model using Global Temperature Data," in *International Conference on Computing Methodologies and Communication (ICCMC)*, 2022.

[3] V. P. and F. M. D. Rao, D. Nandi, F. Pérez-Fontán, "Long Term Prediction of Rain Rate and Attenuation using ANN and RNN Algorithms," in *IEEE International India Geoscience and Remote Sensing Symposium (InGARSS)*, 2021.

[4] S. S. S. and M. A. A. R. Hatem M. Noaman, "Enhancing recurrent neural network-based language models by word tokenization," *Hum. Cent. Comput. Inf. Sci.*, vol. 8, no. 12, 2018.

[5] J. L. Kyungmin Lee, Chiyoun Park, Namhoon Kim, "Accelerating recurrent neural network language model based online speech recognition system," *DMC R&D Center, Samsung Electron. Seoul, Korea*, 2018.

[6] Z. Zhang, H. Cheng, and T. Yang, “A recurrent neural network framework for flexible and adaptive decision making based on sequence learning,” *Plos Comput. Biol.*, 2020.

[7] Z. He, “Improving LSTM Based Acoustic Model with Dropout Method,” in *International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM)*, 2019.

[8] X. C. and F. P. L. Depeng, X. Jianhua, “A modulation recognition method based on enhanced data representation and convolutional neural network,” *2021 IEEE 4th Adv. Inf. Manag. Commun. Electron. Autom. Control Conf.*, 2021.

[9] X. T. and Z. Z. F. Li, X. Yu, “Short-Term Load Forecasting for an Industrial Park Using LSTM-RNN Considering Energy Storage,” in *Asia Energy and Electrical Engineering Symposium (AEEES)*, 2021.

[10] A. A. and B. Kanisha, “Forecasting stock market price using LSTM-RNN,” in *International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, 2022.

[11] Y. Y. and G. Wang, “Self-Adaptive Genetic Algorithm For Bucket Wheel Reclaimer Real-Parameter Optimization,” *IEEE Access*, vol. 7, 2019.

[12] G. Papazoglou, “Review and Comparison of Genetic Algorithm and Particle Swarm Optimization in the Optimal Power Flow Problem,” *Multidiscip. Digit. Publ. Inst.*, 2023.

[13] C. Lamini, S. Benhlima, and A. Elbekri, “Genetic algorithm based approach for autonomous mobile robot path planning,” *Procedia Comput. Sci.*, vol. 127, pp. 180–189, 2018.

[14] S.-U. P. and S.-K. H. J. -H. Han, D. -J. Choi, “Hyperparameter Optimization for Multi-Layer Data Input Using Genetic Algorithm,” in *International Conference on Industrial Engineering and Applications (ICIEA)*, 2020.

[15] X. Gong, “Optimized layout methods based on optimization algorithms for DPOS,” *Aerosp. Sci. Technol.*, 2019.

[16] A. T. and G. Ünal, “A RNN based time series approach for forecasting turkish electricity load,” *Signal Process. Commun. Appl. Conf.*, vol. 26, 2018.

[17] M. M. H. and M. M. K. S. B. Islam, “Prediction of Stock Market Using Recurrent Neural Network,” in *Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, 2021.

[18] M. S. H. and H. Mahmood, “Short-Term Photovoltaic Power Forecasting Using an LSTM Neural Network and Synthetic Weather Forecast,” *IEEE Access*, 2020.

[19] R. V. R. and K. Subramaniam, “Optimized Wavelet Filter from Genetic Algorithm, for Image Compression,” in *International Conference on Smart Structures and Systems (ICSSS)*, 2020.

[20] H. Chung and K. Shin, “Genetic Algorithm-Optimized Long Short-Term Memory Network for Stock Market Prediction,” *MDPI*, 2018.

.

.