Journal of Applied Informatics and Computing (JAIC)
Vol.9, No.1, February 2025, pp. 23~30
e-ISSN: 2548-6861

Comparative Performance Analysis of Optimization Algorithms in Artificial Neural Networks for Stock Price Prediction

Ekaprana Wijaya[1]*, M. Arief Soeleman[2]*, Pulung Nurtantio Andono[3]*
*Teknik Informatika, Universitas Dian Nuswantoro
ekapranawijaya@gmail.com [1], m.arief.soeleman@dsn.dinus.ac.id [2], pulung@dsn.dinus.ac.id [3]

Article Info

Article history:

Received 2024-11-04
Revised 2024-11-18
Accepted 2024-11-25

Keyword:

Artificial Neural Networks,
Model Parameter,
Performance Evaluation,
Stock Price Forecasting.

ABSTRACT

This study aims to enhance price prediction accuracy using Artificial Neural Networks (ANN) by comparing three optimization methods: Stochastic Gradient Descent (SGD), Adam, and RMSprop. The research employs a systematic approach involving the design, training, and validation of ANN models optimized by these techniques. Performance metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), and R Square are utilized to evaluate the effectiveness of each method. The results indicate that the Adam optimization method outperforms the others, achieving the lowest MSE of 0.0000503 and the lowest MAE of 0.0046, resulting in an impressive R Square value of 0.9989. Adam's superior performance can be attributed to its adaptive learning rate mechanism, which effectively adjusts to the high volatility and noise characteristic of stock price data, enabling the model to converge faster and more accurately. In comparison, SGD produced a higher MSE of 0.0001208 and MAE of 0.0075, while RMSprop yielded an MSE of 0.0000726 and MAE of 0.0059. These findings highlight Adam's ability to significantly enhance the predictive capabilities of ANN, particularly in dynamic and complex datasets, making it a preferred choice for this application. The novelty of this research lies not only in its comparative analysis of various optimization methods within the ANN framework but also in the exploration of unique ANN features and their application to a specific stock price prediction case study, providing deeper insights into the practical implications of optimization strategies. This study lays the groundwork for future research by suggesting the exploration of additional optimization algorithms and more complex neural network architectures to further improve prediction accuracy.

I. INTRODUCTION

The dynamic and fluctuating nature of stock price movements in financial markets presents a significant challenge for investors aiming to make accurate predictions [1]. Stock prices are influenced by a multitude of factors, both internal to the company and external, such as global economic conditions, monetary policies, and market sentiment [2]. These complexities create volatility, making it difficult to forecast stock prices with precision, thus increasing the risk for investors when making decisions [3], [4]. While traditional methods like technical analysis, which leverage price and volume data, have been used to identify trends, they often lack responsiveness to rapid market changes and fail to capture the underlying complexities in stock price behavior [5]. One of the primary challenges in stock price prediction is the non-linear nature of the market, influenced by numerous variables simultaneously. Simple predictive models may be effective in short-term scenarios, but they struggle with long-term trends and extreme volatility [6]. As stock price data continues to grow in volume, the need for a more sophisticated approach to process and interpret this information efficiently becomes critical [7]. Investors often face substantial financial risks if they rely on inaccurate predictions in volatile markets. Moreover, existing forecasting techniques, while useful, often do not fully account for the dynamic market conditions that frequently change due to unpredictable external factors.

To address the challenges of stock price prediction in highly volatile and complex financial markets, the use of advanced machine learning techniques, particularly Neural Networks (ANN), offers a robust solution [8], [9]. Artificial Neural Networks, known for their ability to model non-linear relationships and learn from vast amounts of historical data, can effectively capture the intricate patterns in stock price movements. By leveraging techniques such as Stochastic Gradient Descent (SGD), Adam, and RMSprop, Neural Networks can fine-tune predictive models, making them more adaptive to changing market conditions [10]. These algorithms help the ANN learn more efficiently by adjusting the model's parameters to minimize error, thus improving prediction accuracy.

Nti et al. [11] aimed to improve stock price prediction accuracy by integrating data from six heterogeneous sources using a hybrid deep learning framework, IKN-ConvLSTM. The approach combined DNN for feature selection and LSTM networks for prediction refinement. Using stock data from the Ghana Stock Exchange (2017-2020), the model achieved a prediction accuracy of 98.31%, along with a specificity of 99.75%, sensitivity of 89.39%, and F-score of 96.72%, demonstrating the effectiveness of multi-source data fusion for accurate stock market predictions.

Previous study by Kamalov [12] investigates the prediction of significant changes in stock prices using machine learning algorithms, a topic that has received less attention compared to predicting actual asset prices or price direction. The study constructs and evaluates three neural network models—multilayer perceptron, convolutional neural network, and long short-term memory network—while also employing random forest and relative strength index methods as benchmarks. Analyzing ten years of daily stock price data from four major U.S. public companies, the results indicate that significant stock price changes can be predicted with a high level of accuracy, outperforming existing studies focused on forecasting price direction.

Yu et al. [13] explore the complexities of predicting financial data trends, addressing the nonlinear and time-dependent nature of this challenge due to the intricate, incomplete, and fuzzy information inherent in financial activities. The study employs a deep neural network (DNN) model, utilizing the time series phase-space reconstruction (PSR) method to analyze financial product price data as a one-dimensional series derived from a chaotic system. This DNN-based prediction model, which incorporates long- and short-term memory networks (LSTMs), is applied to forecast stock prices across various indices and time periods.

This study employs Artificial Neural Networks (ANN) as the foundational model for predicting stock prices, where various optimization algorithms are applied and their performance compared in terms of accuracy. The dataset used consists of historical stock prices from the LQ45 index. Optimization algorithms such as SGD, Adam, RMSprop are tested on the neural network to assess their impact on prediction accuracy. Each algorithm is evaluated based on accuracy metrics and the loss function to determine how effectively they accelerate convergence and improve prediction accuracy. The results of this research are expected to contribute to identifying the most suitable optimization algorithm for stock price forecasting using ANN.

II. PROPOSED METHOD

The proposed scheme is outlined in Figure 1, which illustrates the various components and steps involved in the process. This visual representation helps clarify how the different elements interact and contribute to the overall objective of the study.

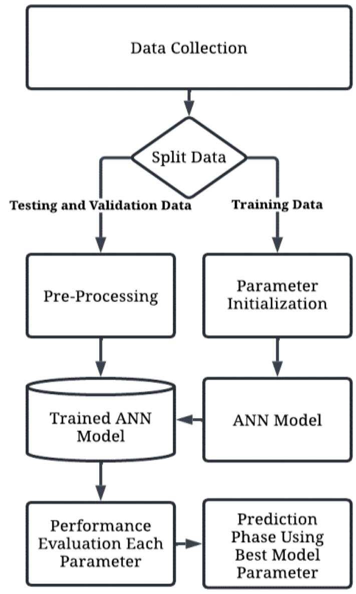


Figure 1. Flow of Proposed Scheme

A. Data Collection and Spliting Data

In this stage, the dataset utilized in this study was sourced from the BMRI index, consisting of historical stock price data for one year, specifically from January 1, 2020, to December 31, 2020 [14]. This dataset includes several attributes: date, opening price, highest price, lowest price, closing price, adjusted closing price, and volume. The "Open" attribute indicates the stock's opening price, "High" denotes the maximum price reached during the day, "Low" shows the minimum price, "Close" represents the closing price, "Volume" indicates the number of transactions for that day, and "Adj Close" refers to the adjusted closing price.

The dataset is divided into two segments: features and target. The features consist of the data used as input to predict the desired target value. In this case, the dataset is split such that the closing prices from the previous 30 days serve as features, while the closing price on the 31st day is designated as the target. This means each feature set will comprise 30 closing prices, and the target represents the closing price on the 31st day. This approach allows for the development of a model capable of predicting the closing price for the next day based on historical patterns from the preceding 30 days. By utilizing historical data as features, the model learns to identify patterns and trends in stock price movements, facilitating accurate predictions for the following day. The results of this data splitting can be observed in Table 1.

TABEL I
SAMPLE SPLITED DATA

Sample Data	Feature	Target
1st Dataset	0.00346691249510021, 0.00346691249510021, 0.00260018490931641, 0.00346691249510021, 0.00346691249510021, 0.00346691249510021, 0.00346691249510021, 0.00346691249510021, 0.00260018490931641, 0.00260018490931641, 0.00173345732353262, 0.00173345732353262, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0, 0, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.00346691249510021, 0.00346691249510021	0.003467
2nd Dataset	0.00346691249510021, 0.00260018490931641, 0.00346691249510021, 0.00346691249510021, 0.00346691249510021, 0.00346691249510021, 0.00346691249510021, 0.00260018490931641, 0.00260018490931641, 0.00173345732353262, 0.00173345732353262, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0, 0, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.00346691249510021, 0.00346691249510021, 0.00346691249510021	0.003467
3rd Dataset	0.00260018490931641, 0.00346691249510021, 0.00346691249510021, 0.00346691249510021, 0.00346691249510021, 0.00346691249510021, 0.00260018490931641, 0.00260018490931641, 0.00173345732353262, 0.00173345732353262, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0, 0, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.00346691249510021, 0.00346691249510021, 0.00346691249510021, 0.00346691249510021	0.003467
4th Dataset	0.00346691249510021, 0.00346691249510021, 0.00346691249510021, 0.00346691249510021, 0.00346691249510021, 0.00260018490931641, 0.00260018490931641, 0.00173345732353262, 0.00173345732353262, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0, 0, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.00346691249510021, 0.00346691249510021, 0.00346691249510021, 0.00346691249510021, 0.00346691249510021	0.003467
5th Dataset	0.00346691249510021, 0.00346691249510021, 0.00346691249510021, 0.00346691249510021, 0.00260018490931641, 0.00260018490931641, 0.00173345732353262, 0.00173345732353262, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0, 0, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.000866729737748824, 0.00346691249510021, 0.00346691249510021, 0.00346691249510021, 0.00346691249510021, 0.00346691249510021, 0.00346691249510021	0.003467

B. Pre-processing

Data preprocessing is a critical step in model development, particularly for training and testing processes [15]. In this analysis, preprocessing involves removing invalid or null data entries from the entire dataset. Initially, the total number of available data rows was 5,132. After eliminating invalid or null entries, the remaining number of data rows was 5,120. Subsequently, irrelevant columns were removed, retaining only the 'date' and 'close' columns. This decision was made because the analysis employs a time series approach, focusing solely on these two columns. The 'date' column represents the trading date, while the 'close' column indicates the closing price of PT Bank Mandiri Tbk (BMRI) at the end of each trading day. The removal of invalid data ensures that no erroneous data interfere with the model during the training and testing phases. Additionally, the elimination of irrelevant columns enables the model to concentrate on the most significant features, namely the trading date and closing price, in time series analysis and predictions. Following these processes, the data undergoes standardization to narrow the range of nominal values between 0 and 1. This step is essential for enhancing the performance and stability of the model during the learning process. The standardization process is executed using the following Equation:

$$Xscaled = \frac{X - Xmin}{Xmax - Xmin} \tag{1}$$

This formula ensures that each value X in the dataset is converted to a value within the range $[0, 1]$, where X_{min} is transformed to 0 and X_{max} is transformed to 1. This process maintains the relative proportions among the values in the dataset during scaling. Table 2 below presents a sample of the data after undergoing the preprocessing stage.

TABEL II
PRE-PROCESSED DATA

Date	Close
7/14/2003	0.003466912
7/15/2003	0.003466912

7/16/2003	0.002600185
7/17/2003	0.003466912
7/18/2003	0.003466912
7/21/2003	0.003466912
7/22/2003	0.003466912
7/23/2003	0.003466912
7/24/2003	0.002600185
7/25/2003	0.002600185
7/28/2003	0.001733457
7/29/2003	0.001733457
7/30/2003	0.00086673
7/31/2003	0.00086673
8/1/2003	0.00086673
8/4/2003	0.00086673
8/5/2003	0
8/6/2003	0
8/7/2003	0.00086673
8/8/2003	0.00086673
8/11/2003	0.00086673
8/12/2003	0.00086673
8/13/2003	0.00086673
8/14/2003	0.00086673
8/15/2003	0.00086673
8/18/2003	0.00086673
8/19/2003	0.00086673
8/20/2003	0.00086673
8/21/2003	0.003466912
8/22/2003	0.003466912

C. *Artificial Neural Networks*

Artificial neural networks are computational models vaguely based on the neural structure of the animal brain [16], [17]. These networks have been built especially to identify and extract knowledge from raw data using a methodology named training. ANNs comprise interconnected collective entities of artificial neurons that process inputs to create outputs [18], [19]. Every connection between neurons has an assigned weight, which changes with each consecutive step of learning, allowing the network to minimize error and therefore make more accurate predictions [20], [21].

The general architecture of the ANN includes layers of neurons, generally the following three: input layers, hidden layers, and output layers. The input layer feeds in the initial data, while the hidden layers carry on with the calculation and feature extraction. The final prediction or classification, among all those learned patterns, is done at the output layer.

1) *Single Layer Network*

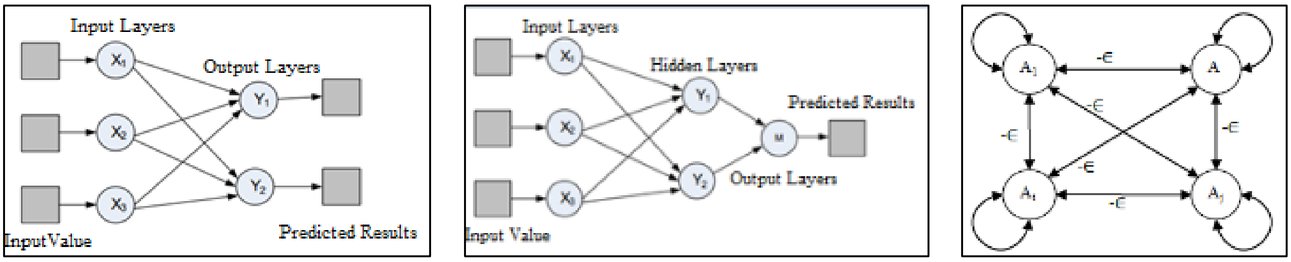
A single-layer network consists of one input layer and one output layer. Each neuron in the input layer is fully connected to each neuron in the output layer. This type of network processes inputs directly into outputs without the need for hidden layers. Notable examples of algorithms that utilize this method include ADALINE, Hopfield networks, and Perceptron models. The architecture of a single-layer network is illustrated in Figure 2 (a).

2) *Multi Layer Network*

Multi-layer networks are characterized by having three types of layers: an input layer, an output layer, and one or more hidden layers. This architecture enables the network to tackle more complex problems compared to single-layer networks. However, training processes may require significantly more time. Common algorithms that implement this approach include MADALINE, Backpropagation, and Neocognitron. The architecture of a multi-layer network is shown in Figure 2 (b).

3) *Competitive Layer Network*

In competitive layer networks, a set of neurons competes to become active. An example of an algorithm that employs this method is Learning Vector Quantization (LVQ). These networks are often used for pattern recognition and classification tasks. The architecture of a competitive-layer network is illustrated in Figure 2 (c).



(a) Single Layer (b) Multi Layer (c) Competitive Layer

Figure 2. ANN Layers

D. *Performance Evaluation*

Evaluation of performance is a very important process in the determination of the performance of any ANN model concerning effectiveness. Precisely, a model will predict the outcomes based on the degree of the training data. Some of the popularly used metrics for the evaluation of the regression models' performance include Mean Squared Error, Mean Absolute Error, and R-squared. MSE maps the average of the squared differences between predictions and actual values [22]. This is one such measure that denotes accuracy about the model in its predictions-the lesser the value, the better the performance of the model.

While Mean Absolute Error simply deals with the average value of the absolute differences between the predictions and actual values [23]. It turns out this is way more intuitive than the MSE, since here the score is linear and can be interpreted as some sort of average error in the same unit as the forecasted values. R-squared, or R^2, refers to the statistical value which calculates a percentage of variation in a dependent variable explained by independent variables in a regression model [23]. The R^2 values vary within the range from 0 to 1. A close-to-1 number means that a big part of the dependent variable is explained by the model. The equation based on performance evaluation can be seen in eq (2) – (4). Where, n is total number of observations. yi is the the actual value. $y^\wedge i$ the predicted value.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(yi - y^\wedge i)^2 \tag{2}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}(yi - y^\wedge i) \tag{3}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(yi - y^\wedge i)^2}{\sum_{i=1}^{n}(yi - y\sim i)^2} \tag{4}$$

III. Results And Discussion

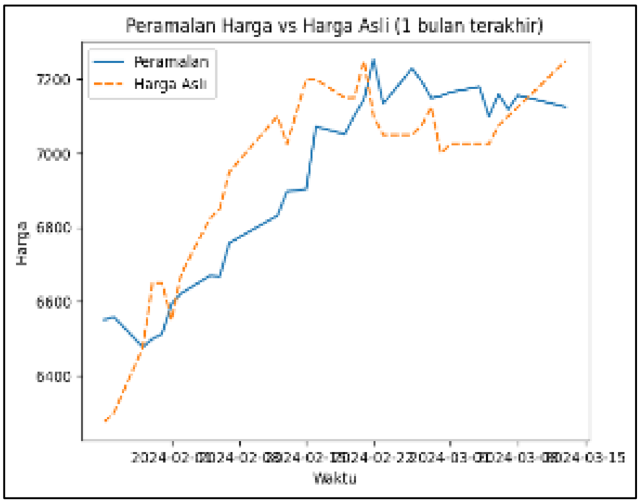
In this study, multiple training experiments were conducted using seven different optimization parameters to evaluate their impact on the performance of the Artificial Neural Network (ANN) model. The optimization algorithms tested include Stochastic Gradient Descent (SGD), Adam (Adaptive Moment Estimation), RMSprop (Root Mean Square Propagation).

The results obtained from each optimization experiment are analyzed through various performance metrics, including Mean Squared Error (MSE), Mean Absolute Error (MAE), and R-Squared (R^2) values. These metrics provide insight into the effectiveness of each optimization strategy in minimizing error and improving predictive accuracy. The performance results are presented in tabular form and graph image, allowing for a straightforward comparison of the metrics across different algorithms. Additionally, graphical representations illustrate the trends and performance differences of the optimization algorithms throughout the training process.

A. Stochastic Gradient Descent (SGD)

SGD demonstrates a reasonable ability to capture long-term price trends, showing a close correlation with actual prices over several years. However, the graph indicates that the predictions experience notable fluctuations in the short term, leading to some divergence from actual prices in recent months. This suggests that while SGD can effectively follow general trends, it may struggle with the volatility of short-term price movements, indicating potential sensitivity to recent data changes. Results graph of SGD parameter can be seen in Figure 3.

(a) Predicted Price vs Actual Price Per Year



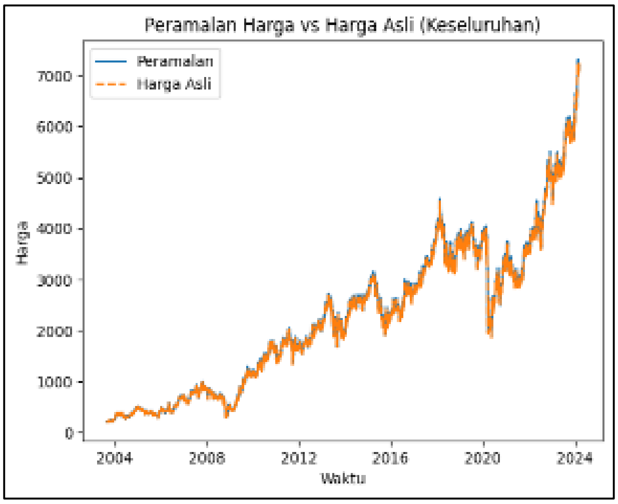
(b) Predicted Price vs Actual Price Per Month

Figure 3. Results of SGD Parameter

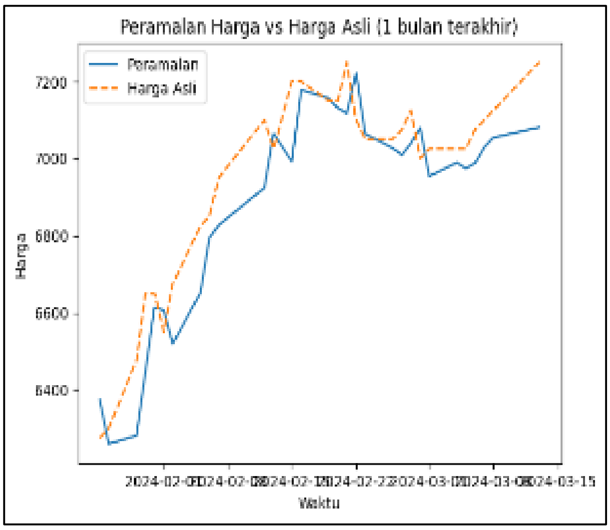
B. Adaptive Moment Estimation (ADAM)

Adam optimization algorithm exhibits strong performance in both long-term and short-term predictions, closely aligning with actual price movements over the years. Its ability to adapt to changing trends is particularly evident in the recent month, where the predicted values remain stable with only minor deviations from actual prices. Results graph of ADAM parameter can be seen in Figure 4.

(a) Predicted Price vs Actual Price Per Year

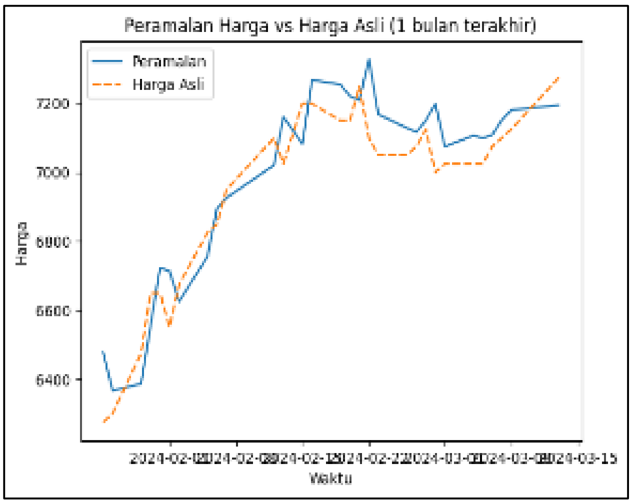


(a) Predicted Price vs Actual Price Per Year



(b) Predicted Price vs Actual Price Per Month

Figure 4. Results of ADAM Parameter



(b) Predicted Price vs Actual Price Per Month

Figure 5. Results of RMSPRops Parameter

C. *Root Mean Square Propagations (RMSProps)*

RMSprop proves to be a robust optimization method, effectively capturing the overall trend of prices similar to Adam. It excels at addressing issues related to vanishing or exploding gradients, which enhances its reliability for long-term forecasting. In recent predictions, RMSprop demonstrates a strong alignment with actual prices, indicating its effectiveness in adapting to short-term price dynamics. Results graph of RMSProps parameter can be seen in Figure 5.

D. *Performance Evaluation*

Table III highlights the performance evaluation of three optimization algorithms: Adam, SGD, and RMSprop. Among these, Adam demonstrates the best performance, achieving the lowest Mean Squared Error (MSE) of 5.03e-05 and Mean Absolute Error (MAE) of 0.00460, along with the highest R-Squared (R^2) value of 0.9989, indicating an excellent fit to the data. This superior performance can be attributed to Adam's adaptive learning rate mechanism, which combines the benefits of momentum and per-parameter learning rate adjustments, enabling it to handle the high volatility and noise often present in stock price datasets. RMSprop follows closely, with an MSE of 7.26e-05 and MAE of 0.00589, resulting in an R^2 of 0.9984. RMSprop's strength lies in its ability to maintain a stable learning rate for frequently updated parameters, making it effective for this context. SGD, while effective, has slightly higher error metrics, with an MSE of 0.0001208 and MAE of 0.00752, resulting in an R^2 of 0.9973, potentially due to its fixed learning rate, which can struggle with dynamic and noisy data. Overall, Adam is the most effective optimization method among the three, providing the best accuracy and data fitting.

TABEL III
PERFORMANCE EVALUATION

Optimalization Parameter	MSE	MAE	R^2
SGD	0.00012	0.00752	0.99729
ADAM	0.00005	0.00459	0.99887
RMSProps	0.00007	0.00589	0.99836

IV. CONCLUSIONS

In this study, we proposed a novel approach for predicting prices using various optimization methods in artificial neural networks (ANN). The methods evaluated include Stochastic Gradient Descent (SGD), Adam, and RMSprop. Each optimization technique was subjected to rigorous training and validation processes, resulting in significant insights into their performance metrics. The results, demonstrate that the Adam optimization method achieved the lowest Mean Squared Error (MSE) of 0.0000503 and the lowest Mean Absolute Error (MAE) of 0.0046, along with the highest R Square value of 0.9989. This indicates that the Adam optimizer effectively minimizes prediction error, making it highly reliable for this particular application. Conversely, while the SGD method performed well with an MSE of 0.0001208 and a MAE of 0.0075, it lagged behind Adam in terms of predictive accuracy. RMSprop also showed promising results, with an MSE of 0.0000726 and a MAE of 0.0059, highlighting its effectiveness in handling non-stationary objectives.

These findings suggest that utilizing advanced optimization techniques like Adam can significantly enhance the performance of ANN models in price prediction tasks. For future research, we recommend exploring additional optimization algorithms, such as Nadam and Adagrad, and investigating their applicability in different domains. Furthermore, expanding the dataset and incorporating more complex neural network architectures could yield even more accurate predictions, allowing for broader applicability in real-world scenarios.

REFERENCES

[1] O. Bustos and A. Pomares-Quimbaya, "Stock market movement forecast: A Systematic review," *Expert Syst Appl*, vol. 156, p. 113464, Oct. 2020, doi: 10.1016/j.eswa.2020.113464.

[2] A. Thakkar and K. Chaudhari, "A comprehensive survey on deep neural networks for stock market: The need, challenges, and future directions," *Expert Syst Appl*, vol. 177, p. 114800, Sep. 2021, doi: 10.1016/j.eswa.2021.114800.

[3] D. Sheth and M. Shah, "Predicting stock market using machine learning: best and accurate way to know future stock prices," *International Journal of System Assurance Engineering and Management*, vol. 14, no. 1, pp. 1–18, Feb. 2023, doi: 10.1007/s13198-022-01811-1.

[4] G. Kumar, S. Jain, and U. P. Singh, "Stock Market Forecasting Using Computational Intelligence: A Survey," *Archives of Computational Methods in Engineering*, vol. 28, no. 3, pp. 1069–1101, May 2021, doi: 10.1007/s11831-020-09413-5.

[5] R. Bhowmik and S. Wang, "Stock Market Volatility and Return Analysis: A Systematic Literature Review," *Entropy*, vol. 22, no. 5, p. 522, May 2020, doi: 10.3390/e22050522.

[6] A. Thakkar and K. Chaudhari, "A Comprehensive Survey on Portfolio Optimization, Stock Price and Trend Prediction Using Particle Swarm Optimization," *Archives of Computational Methods in Engineering*, vol. 28, no. 4, pp. 2133–2164, Jun. 2021, doi: 10.1007/s11831-020-09448-8.

[7] W. Yu, C. Y. Wong, R. Chavez, and M. A. Jacobs, "Integrating big data analytics into supply chain finance: The roles of information processing and data-driven culture," *Int J Prod Econ*, vol. 236, p. 108135, Jun. 2021, doi: 10.1016/j.ijpe.2021.108135.

[8] A. Setiawan, A. S. Prabowo, and E. Y. Puspaningrum, "Handwriting Character Recognition Javanese Letters Based on Artificial Neural Network," 2019.

[9] M. Shorfuzzaman, M. Masud, H. Alhumyani, D. Anand, and A. Singh, "Artificial Neural Network-Based Deep Learning Model for COVID-19 Patient Detection Using X-Ray Chest Images," *J Healthc Eng*, vol. 2021, 2021, doi: 10.1155/2021/5513679.

[10] A. Kumar, S. Sarkar, and C. Pradhan, "Malaria Disease Detection Using CNN Technique with SGD, RMSprop and ADAM Optimizers," in *Deep Learning Techniques for Biomedical and Health Informatics*, S. Dash, B. R. Acharya, M. Mittal, A. Abraham, and A. Kelemen, Eds., Cham: Springer International Publishing, 2020, pp. 211–230. doi: 10.1007/978-3-030-33966-1_11.

[11] I. K. Nti, A. F. Adekoya, and B. A. Weyori, "A novel multi-source information-fusion predictive framework based on deep neural networks for accuracy enhancement in stock market prediction," *J Big Data*, vol. 8, no. 1, p. 17, Dec. 2021, doi: 10.1186/s40537-020-00400-y.

[12] F. Kamalov, "Forecasting significant stock price changes using neural networks," *Neural Comput Appl*, vol. 32, no. 23, pp. 17655–17667, Dec. 2020, doi: 10.1007/s00521-020-04942-3.

[13] P. Yu and X. Yan, "Stock price prediction based on deep neural networks," *Neural Comput Appl*, vol. 32, no. 6, pp. 1609–1628, Mar. 2020, doi: 10.1007/s00521-019-04212-x.

[14] "BMRI.JK Datasets." Accessed: Oct. 11, 2024. [Online]. Available: https://sg.finance.yahoo.com/quote/BMRI.JK/

[15] Q. A. Putra, C. A. Sari, E. H. Rachmawanto, N. R. D. Cahyo, E. Mulyanto, and M. A. Alkhafaji, "White Bread Mold Detection using K-Means Clustering Based on Grey Level Co-Occurrence Matrix and Region of Interest," in *2023 International Seminar on Application for Technology of Information and Communication (iSemantic)*, 2023, pp. 376–381. doi: 10.1109/iSemantic59612.2023.10295369.

[16] N. R. D. Cahyo and M. M. I. Al-Ghiffary, "An Image Processing Study: Image Enhancement, Image Segmentation, and Image Classification using Milkfish Freshness Images," *IJECAR) International Journal of Engineering Computing Advanced Research*, vol. 1, no. 1, pp. 11–22, 2024.

[17] F. Farhan, C. A. Sari, E. H. Rachmawanto, and N. R. D. Cahyo, "Mangrove Tree Species Classification Based on Leaf, Stem, and Seed Characteristics Using Convolutional Neural Networks with K-Folds Cross Validation Optimalization," *Advance Sustainable Science Engineering and Technology*, vol. 5, no. 3, p. 02303011, Oct. 2023, doi: 10.26877/asset.v5i3.17188.

[18] I. P. Kamila, C. A. Sari, E. H. Rachmawanto, and N. R. D. Cahyo, "A Good Evaluation Based on Confusion Matrix for Lung Diseases Classification using Convolutional Neural Networks," *Advance Sustainable Science, Engineering and Technology*, vol. 6, no. 1, p. 0240102, Dec. 2023, doi: 10.26877/asset.v6i1.17330.

[19] M. M. I. Al-Ghiffary, N. R. D. Cahyo, E. H. Rachmawanto, C. Irawan, and N. Hendriyanto, "Adaptive deep learning based on FaceNet convolutional neural network for facial expression recognition," *Journal of Soft Computing*, vol. 05, no. 03, pp. 271–280, 2024, doi: https://doi.org/10.52465/joscex.v5i3.450.

[20] N. R. D. Cahyo, C. A. Sari, E. H. Rachmawanto, C. Jatmoko, R. R. A. Al-Jawry, and M. A. Alkhafaji, "A Comparison of Multi Class Support Vector Machine vs Deep Convolutional Neural Network for Brain Tumor Classification," in *2023 International Seminar on Application for Technology of Information and Communication*

(iSemantic), IEEE, Sep. 2023, pp. 358–363. doi: 10.1109/iSemantic59612.2023.10295336.

[21] M. M. I. Al-Ghiffary, C. A. Sari, E. H. Rachmawanto, N. M. Yacoob, N. R. D. Cahyo, and R. R. Ali, "Milkfish Freshness Classification Using Convolutional Neural Networks Based on Resnet50 Architecture," *Advance Sustainable Science Engineering and Technology*, vol. 5, no. 3, p. 0230304, Oct. 2023, doi: 10.26877/asset.v5i3.17017.

[22] U. Sara, M. Akter, and M. S. Uddin, "Image Quality Assessment through FSIM, SSIM, MSE and PSNR—A Comparative Study," *Journal of Computer and Communications*, vol. 07, no. 03, pp. 8–18, 2019, doi: 10.4236/jcc.2019.73002.

[23] D. Chicco, M. J. Warrens, and G. Jurman, "The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation," *PeerJ Comput Sci*, vol. 7, p. e623, Jul. 2021, doi: 10.7717/peerj-cs.623.