Vokasi Unesa Bulletin of Engineering, Technology and Applied Science (VUBETA)
https://journal.unesa.ac.id/index.php/vubeta
Vol. 3, No. 1, 2026, pp. 64~79 DOI: 10.26740/vubeta.v3i1.45130
ISSN: 3064-0768

Stock Price Forecasting Using LSTM with Cross-Validation

Rifki Ainul Yaqin[1,2]*, Muhammad Iqbal Anshori[1,2], Reddis Angel[1,2], Ignatius Wiseto Prasetyo Agung[1,2], Toni Arifin[1,2], Erfian Junianto[1,2]
[1] Information Engineering Study Program, Faculty of Information Technology, Universitas Adhirajasa Reswara Sanjaya (ARS University), Bandung, Indonesia
[2] ARS Digital Research and Innovation (ADRI), Universitas Adhirajasa Reswara Sanjaya (ARS University), Bandung, Indonesia

Article Info

Article history:

Received August 14, 2025
Revised October 15, 2025
Accepted December 15, 2025

Keywords:

Stock Price Forecasting
LSTM (Long Short-Term Memory)
Cross-Validation
Deep Learning
Sequence Forecasting

ABSTRACT

Stock price forecasting is highly challenging due to the market's nonlinear, volatile nature, which is influenced by complex economic and behavioral factors. Traditional statistical models and many machine learning approaches often suffer from overfitting and limited generalizability. This study examines the effectiveness of Long Short-Term Memory (LSTM) networks combined with k-Fold Cross-Validation as a lightweight yet robust alternative. Unlike Transformer-based models, which require extensive computational resources, LSTM offers a more resource-efficient solution while effectively capturing temporal dependencies in financial time series. Experiments were conducted on six U.S. stocks (LW, LKQ, IPG, MGM, RL, and CAG) across 1,000 training epochs, using one to two LSTM layers (64–128 hidden units) with the Adam optimizer. Model performance was evaluated using RMSE, MAE, and R^2 under k-Fold Cross-Validation and compared against Split Validation from prior studies. Results show that k-Fold consistently produced lower error values, confirming its reliability for stable performance estimation. Notably, models using Close-only input achieved lower RMSE and MAE than those with additional indicators (MA200, stochastic), which primarily improved R^2. This indicates that feature simplicity, combined with robust preprocessing and validation, can outperform more complex inputs in short-term forecasting. In conclusion, integrating LSTM with k-Fold Cross-Validation provides a practical and efficient framework for stock prediction, particularly in resource-constrained settings. However, the findings are limited to specific stocks and indicators. Future work should extend the approach to broader markets, incorporate macroeconomic or sentiment-based features, and explore hybrid architectures to enhance predictive performance further.

1. INTRODUCTION

Stock price movements are notoriously difficult to predict due to complex factors, including global economic conditions, unemployment rates, monetary policy, natural disasters, and public health crises. These uncertainties drive market participants to seek methods to maximize profits and minimize risk through comprehensive market analysis [1]. The stock market plays a vital role in the economy by providing liquidity, supporting diversification, optimizing resource allocation, and reducing information and transaction costs. Numerous studies have also demonstrated a positive relationship between stock market development and a country's economic growth [2]. As financial instruments, stocks represent partial ownership in a company, granting rights to profits and participation in key decisions, though they remain vulnerable to price fluctuations driven by market dynamics [3]. In response to these challenges, technological advancements such as artificial intelligence and big data have fueled significant interest in stock price prediction across both industry and academia. Algorithms such as decision trees, Support Vector Machines (SVMs), and Long Short-Term Memory (LSTM) networks have been widely adopted, with LSTM proving particularly effective for processing volatile time-series data [4]. Traditional models are increasingly viewed as insufficient, making big data and

*Corresponding Author
Email: rifkiainul17@gmail.com

neural network-based approaches more appealing [5]. Meanwhile, the Efficient Market Hypothesis (EMH) remains a subject of debate, and technical analysis has emerged as a primary method for leveraging historical market patterns in stock price forecasting [6][7].

Stock price prediction poses a significant challenge due to the highly dynamic, nonlinear nature of the market, which is influenced by economic, political, and often irrational investor sentiment [8]. Traditional time series analysis methods, such as ARIMA and Exponential Smoothing, have long been employed in financial forecasting. However, these approaches are limited in their ability to handle the inherent complexity and high volatility of financial data [9][10]. In addition, machine learning techniques such as Support Vector Machines, Random Forests, and Artificial Neural Networks have been widely applied, yet they often encounter such problems as overfitting, high data dimensionality, and reliance on manual feature engineering [11][12]. To overcome these limitations, researchers have increasingly turned to deep learning-based approaches, particularly Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) architectures. These models possess capability to capture long-term patterns in time-series data and have demonstrated promising results in stock price prediction tasks [13]-[15]. Nevertheless, prediction accuracy remains a critical concern. Therefore, more robust model evaluation strategies, such as Cross-Validation, are necessary to enhance both the accuracy and generalizability of models when applied to the complexities of financial market data.

Recent studies further highlight the strengths of LSTM-based forecasting when combined with appropriate validation and comprehensive evaluation metrics. For instance, Varadharajan *et al.* (2024) employed an LSTM-RNN model to predict Amazon's daily closing prices using an 80–20 data split. Their evaluation included RMSE, MAE, and MAPE, achieving results of 2.84, 2.043, and 1.510, respectively, under optimal hyperparameter tuning and dropout regularization, which helped mitigate overfitting [16]. Similarly, recent work by Chang et al. (2024) demonstrated that integrating multiple performance metrics, such as RMSE, MAE, and R^2, provides a more robust understanding of predictive accuracy and generalization in volatile financial time series. These findings underscore the importance of diverse metrics and systematic validation in LSTM-based stock price forecasting [17].

While these studies demonstrate the importance of rigorous validation and diverse performance metrics, Agung *et al.* (2025) reveal several notable limitations in their methodology. First, the study used a static data-splitting method, allocating 96% to training and 4% to testing. While straightforward, this approach is overly limited in the context of stock market forecasting, as it ignores temporal variability in dynamic and non-stationary data, increasing the risk of overfitting and leading to an optimistic bias in performance evaluation. Second, the model evaluation focused solely on RMSE and average profit, omitting complementary metrics such as MAE and R^2, which would provide a more comprehensive assessment of accuracy and error distribution. Third, the study placed excessive emphasis on profit outcomes as the primary measure of success. Although profit is practically relevant, it is susceptible to specific market conditions and does not adequately reflect the model's ability to generalize across unseen data or varying scenarios. Finally, the generalization capability of the proposed framework remains weak, as the authors themselves acknowledged the need for future validation using Cross-Validation methods to strengthen the reliability of their results. These gaps highlight the need for methodological improvements, particularly the integration of robust time-series Cross-Validation and multiple performance metrics, to ensure both accuracy and resilience in LSTM-based stock price forecasting [18].

To address these limitations, this study employs k-Fold Cross-Validation to enhance the reliability of LSTM models for stock price forecasting. Cross-validation helps mitigate the risk of overfitting by partitioning the dataset into multiple rotating training and testing subsets, thereby producing more stable evaluation results and improving the model's generalization to unseen data [19]. Furthermore, the integration of deep learning models such as LSTM with robust validation techniques has been shown to yield more accurate predictions for complex, nonlinear time series such as financial data [20][21]. The application of grid search Cross-Validation also plays a critical role in the hyperparameter tuning process, ensuring optimal performance according to predefined evaluation metrics [22]. Thus, integrating Cross-Validation into deep learning architectures such as LSTM and its variants offers a promising pathway to address the volatility and uncertainty inherent in stock price forecasting [23].

With the validation framework established, it is essential to justify the choice of model architecture. The selection of LSTM in this stock forecasting study is based on its advantages over other models, particularly traditional RNN and Transformers. LSTMs are specifically designed to overcome the vanishing and exploding gradient problems in RNN, enabling them to retain long-term memory, which is crucial for sequential data such as stock prices [24][25]. With their memory cell structure and three main gates (input, forget, and output), LSTM can adaptively decide which information to retain or discard, thus enhancing their ability to learn from complex historical patterns [26]. In contrast, although Transformer-based models are powerful in capturing multivariate interactions, they suffer from quadratic computational complexity, which limits their practicality for long sequences [27]. Therefore, LSTMs remain a mature, stable, and proven architecture for handling long-term dependencies while maintaining computational efficiency, making them a strong baseline and an appropriate choice for forecasting dynamic, highly uncertain stock price movements and time-series data in real-world applications.

In this study, titled “Stock Price Forecasting Using LSTM with Cross-Validation,” the novelty lies in the systematic integration of the LSTM deep learning model with k-Fold Cross-Validation for forecasting stock prices characterized by dynamic and nonlinear behavior. Unlike previous studies, such as Agung *et al.* (2025), which relied solely on a static 96:4 data-splitting approach without accounting for temporal variability, this research offers a more robust and generalizable model evaluation framework [18]. The application of Cross-Validation not only mitigates the risk of overfitting but also yields more stable and representative evaluation results under real-world, fluctuating market conditions. Additionally, the implementation of Cross-Validation for hyperparameter tuning within the LSTM architecture is a further contribution, enhancing both the accuracy and consistency of model performance.

The specific contributions of this study are as follows:

1. Introducing k-Fold Cross-Validation as a systematic validation method for LSTM in stock price forecasting, specifically applied to 10-day ahead predictions.
2. Employing multiple evaluation metrics, including RMSE, MAE, and R^2, to provide a more comprehensive assessment of forecasting performance.
3. Exploring different hidden sizes (64 and 128 units) within the LSTM layers to examine their effects on accuracy and stability.
4. Assessing computational time requirements, offering insights into the trade-off between model complexity and efficiency in practice.

The remainder of this paper is structured as follows: Section 2 details the methodology, Section 3 presents the experimental results and analysis, and Section 4 concludes with key findings and directions for future research.

2. METHOD

The methodology of this study is designed to ensure the robustness and reliability of the proposed approach for forecasting and evaluating model performance. It integrates deep learning techniques with systematic validation strategies and a well-structured experimental workflow. Specifically, it highlights the role of Long Short-Term Memory (LSTM) networks in modeling sequential data, the use of Cross-Validation to minimize bias and enhance generalizability, and an experimental workflow that delineates the end-to-end process from data preparation to training, validation, and testing. Together, these components provide a solid foundation for reliable and accurate results.

2.1. Long Short-Term Memory

Long Short-Term Memory (LSTM) is an advanced architecture of Recurrent Neural Networks (RNNs) specifically designed to address long-term dependency issues in time series data, such as vanishing and exploding gradients. The LSTM architecture introduces a memory cell (cell state) and three primary gates: the input gate, forget gate, and output gate, which selectively regulate the flow of information. The forget gate determines how much information from the previous step is retained, the input gate controls how much new information is added, and the output gate decides which information is passed on to the next time step. This mechanism combines linear operations with non-linear activation functions, such as sigmoid and tanh, allowing the LSTM to filter and retain critical information, thereby improving prediction accuracy. Through internal feedback via the memory cell and hidden state, LSTM networks are capable of maintaining long-term temporal information and managing complex dynamics in time series data more effectively and stably than traditional RNNs [28]-[30]. An illustration of the LSTM architecture is presented in Figure 1.

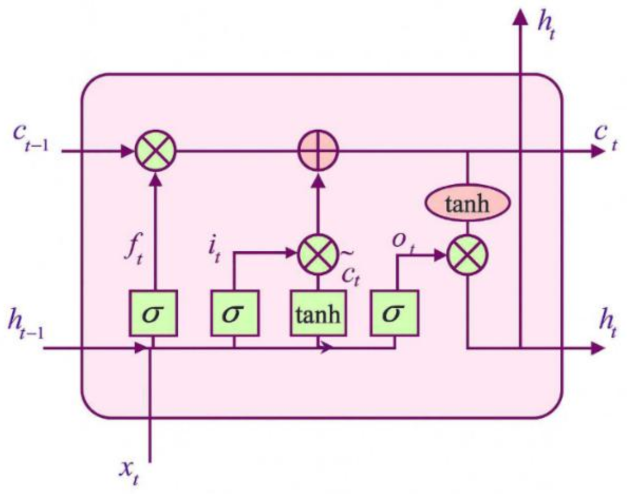


Figure 1. LSTM unit structure [31]

The architecture of Long Short-Term Memory (LSTM) can be formally described through a series of fundamental equations, namely the forget gate Equation 1, input gate Equation 2, candidate cell state Equation 3, cell state update Equation 4, output gate Equation 5, and hidden state Equation 6, each representing a key component that governs the information flow within the LSTM unit.

1. Forget Gate
$$f_t = \sigma(W_{xf} \cdot x_t + W_{hf} \cdot h_{t-1} + b_f) \quad (1)$$
2. Input Gate
$$i_t = \sigma(W_{xi} \cdot x_t + W_{hi} \cdot h_{t-1} + b_i) \quad (2)$$
3. Candidate Cell State
$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \quad (3)$$
4. Cell State Update
$$\tilde{c}_t = \tanh(W_{xc} \cdot x_t + W_{hc} \odot h_{t-1} + b_c) \quad (4)$$
5. Output Gate
$$o_t = \sigma(W_{xo} \cdot x_t + W_{ho} \cdot h_{t-1} + b_o) \quad (5)$$
6. Hidden State (Output)
$$h_t = o_t \odot \tanh(c_t) \quad (6)$$

- Forget gate (Equation 1): determines the portion of information from the previous cell state that should be retained.
- Input gate (Equation 2): regulates the amount of new information incorporated into the current cell state.
- Candidate cell state (Equation 3-4): represents the potential new content generated through a non-linear transformation (tanh) of the input and previous hidden state.
- Output gate (Equation 5): controls which part of the updated cell state is exposed to the hidden state.
- Hidden state (Equation 6): produces the final output of the LSTM unit and serves as input for the next time step.

Here, f_t denotes the proportion of past information retained after the forget gate, i_t indicates the contribution of new input, $\tilde{c}_t$ is the candidate cell state, and c_t is the updated cell state. The sigmoid function (σ) is used for gating operations, while $\odot$ denotes the element-wise (Hadamard) product. x_t refers to the current input vector, h_{t-1} represents the hidden state from the previous step, $W_{xf}, W_{xi}, W_{xc}, W_{xo}$ are input weight matrices, $W_{hf}, W_{hi}, W_{hc}, W_{ho}$ are recurrent weight matrices associated with the hidden state, and b_f, b_i, b_c, b_o are the respective bias vectors for each gate [31].

2.2. Cross-Validation

In many cases, training and testing a predictive algorithm are performed on a single dataset, which may lead to biased performance metrics. Cross-validation (CV) techniques address this limitation by enabling the algorithm to be trained, validated, and tested on multiple subsets, or folds, of the data [32]. In both machine learning and deep learning, standard holdout methods, in which data is split into training, validation, and test sets, and CV are widely used to evaluate model performance [33]. The core principle of CV is to partition the dataset into several folds, with each fold serving in turn as a training and a test set. This process is iterated across different partition configurations to generate more reliable performance estimates and to reduce evaluation bias and the risk of overfitting [34].

CV serves three primary purposes: evaluating a model's generalization capability, selecting the most suitable algorithm among candidates, and supporting hyperparameter optimization [35]. The number of folds selected influences the bias-variance trade-off: increasing the number of folds generally reduces bias but raises variance, whereas fewer folds lower variance but increase bias [36]. As a model evaluation strategy, CV has been employed for decades, typically by splitting available data into separate portions: one for model development and the other for performance testing on unseen data [37]. This iterative and systematic approach enhances the robustness of model evaluation, making it a preferred method for performance assessment in predictive modeling.

2.3. Workflow Experiment

In this experiment, a series of steps is conducted within the experimental workflow. The experiment consists of several stages: dataset selection, data preprocessing, feature selection, data preparation, k-fold cross-validation, LSTM configuration, model training, model evaluation, and results visualization. The complete process is illustrated in Figure 2.

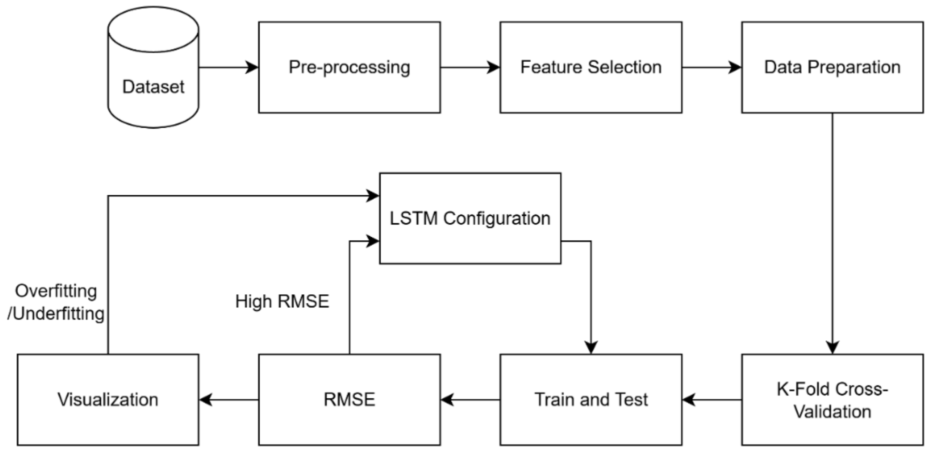


Figure 2. Experimental Workflow

2.3.1. LSTM Configuration

In this experiment, we implemented a model configuration with 1 or 2 Long Short-Term Memory (LSTM) layers, each with 64 or 128 hidden units. The model was designed with a single output unit and a dropout rate of 0.2 applied to each LSTM layer to mitigate overfitting. Training was conducted over 1000 epochs using the Adam optimizer with a learning rate of 0.001 and a weight decay of 0.00001, to ensure faster convergence and improved generalization. Furthermore, a 10-fold Cross-Validation approach was employed with shuffling disabled to preserve the temporal order of the data, thereby providing a robust and reliable estimate of the model's performance across different data partitions. This configuration was chosen to systematically examine the effects of architectural depth, hidden layer size, and hyperparameter settings on the model's effectiveness in addressing the forecasting task.

2.3.2. Dataset

The dataset employed in this study is identical to that used in the previous research and consists of six selected stocks: LW, LKQ, IPG, MGM, RL, and CAG. The selection of these particular stocks was based on their use in previous studies, enabling a more consistent comparison of the results. Furthermore, the dataset spans March 1, 2022, to March 28, 2023, thereby representing approximately one year of stock market dynamics, including trend variations, price fluctuations, and varying market conditions.

For the feature selection process, we relied on both a literature review and empirical considerations. Features such as Close, 200-day Moving Average (MA 200), and Stochastic Oscillator were selected because prior studies have shown their relevance in capturing stock price dynamics. The MA 200 was computed from the rolling average of the past 200 trading days to represent long-term trends, while the Stochastic Oscillator was calculated from the relative position of the current price within a recent high–low range to capture price momentum. These indicators, together with the raw Close price, were normalized and used as multivariate inputs to the LSTM model. This mechanism ensures that the model receives both raw price information and technical signals, enabling it to learn temporal dependencies more effectively and produce more accurate predictions [18].

2.3.3. Normalization

In the data preparation stage, the dataset was split into training and test sets to ensure that test data was not used in training. Subsequently, data normalization was performed using the Min-Max Scaler, mapping the data to the range [-1, 1]. The Min-Max Scaler was selected because it transforms each feature's original values to a specified interval, such as [0, 1] or [-1, 1], thereby preventing features with larger numerical ranges from disproportionately influencing the learning process. This step is crucial since the dataset often contains features with highly varying scales [38]. Without normalization, features expressed in larger magnitudes, such as trading volume, may disproportionately influence similarity-based calculations, leading to biased or suboptimal model performance.

The impact of normalization on forecasting results is significant. Proper scaling ensures that all features contribute proportionally to the model's learning process, which is especially important when combining multiple technical indicators. Inadequate normalization can lead to misleading results, in which a simple feature, such as the closing price, may outperform a combination of indicators because scale differences less influence it. By applying Min-Max Scaling, the weight of each feature is equalized, enabling the model to evaluate their relative importance more objectively. This improves both the fairness and reliability of the prediction results.

The formula for Min-Max normalization [-1,1] is presented in Equation 7 [39].

$$v = \frac{v - x_{min}}{x_{max} - x_{min}} + x_{min} \tag{7}$$

2.3.4. k-Fold Cross-Validation

An automated k-fold cross-validation approach was used to identify the optimal value of λ. This method involves randomly dividing the dataset into *k* equal-sized folds. During each iteration, the model is trained on *k−1* folds and validated on the remaining fold, ensuring that each subset functions as a validation set exactly once. The procedure is repeated *k* times, and the average validation error is computed for various values of λ. The λ value that minimizes this error is selected as the most suitable parameter for the final model configuration [40]. Additionally, this technique generates out-of-fold predictions, enabling a comprehensive evaluation of the model's generalization performance [41]. In the context of stock forecasting, because the data are time series, their temporal order must be preserved. If shuffle is enabled, the data becomes randomized and no longer reflects the chronological sequence, potentially leaking information from the future into the past. Based on this reasoning, shuffling must be disabled to prevent future information from leaking into the training process. Nevertheless, the k-Fold method remains valid when applied with a time-series split or a blocked k-Fold approach, in which the data is partitioned sequentially by time. With this approach, each fold alternates between training and validation data without violating temporal constraints, thereby maintaining the principles of proper Cross-Validation for time-series data.

2.3.5. Optimizer

During training, we use the Adam optimizer because Adam (Adaptive Moment Estimation) is a widely used deep learning optimization algorithm that adapts the learning rate for each parameter based on the first moment (mean of gradients) and the second moment (variance of gradients) [42]. This approach combines the advantages of AdaGrad and RMSProp, enabling faster convergence while maintaining stable weight updates [43]. In the context of stock price prediction, Adam plays an important role, as stock market data is often complex, non-stationary, and highly volatile. Through its adaptive mechanism, Adam allows the model to learn more efficiently from data with varying feature scales, reducing the risk of overfitting while enhancing prediction accuracy and robustness [44]. Therefore, Adam is highly relevant for improving the performance of stock forecasting models compared to conventional optimization algorithms.

2.3.6. Evaluation Metrics

In evaluating model performance, two primary metrics are employed: namely, RMSE and MAE, along with an additional metric, R^2, which assesses the model's ability to capture market trends. In the context of stock price forecasting, RMSE and MAE are the primary focus as they directly indicate the degree of deviation between predicted and actual prices. Meanwhile, R^2 is reported as a complementary measure, providing insights into the proportion of price variance explained by the model. Although certain stocks may exhibit high R^2 values, this is not considered the primary benchmark. The evaluation focuses primarily on minimizing RMSE and MAE values, as these are more relevant in practical forecasting applications. Accordingly, RMSE and MAE are used as primary metrics to assess price prediction accuracy, whereas R^2 serves as a supplementary indicator of the model's ability to capture market trends.

Among these two metrics, RMSE (Root Mean Square Error) is particularly important because it quantifies the average deviation between predicted and actual values. It is calculated as the square root of the mean of the squared differences between expected and actual values [45]. The formula for RMSE is presented in Equation 8 [46].

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \widehat{x_i})^2} \tag{8}$$

In addition to RMSE, Mean Absolute Error (MAE) is widely employed as a primary metric in regression and forecasting tasks. MAE calculates the average of the absolute differences between the actual and predicted values, providing an intuitive measure of error magnitude regardless of direction. Due to its simplicity and ease of interpretation, MAE is often selected as a primary indicator for evaluating the performance of prediction models, particularly to ensure that forecasting results meet the accuracy requirements for operational decision-making [47][48]. The formula for MAE is presented in Equation 9 [49].

$$MAE = \frac{1}{n}\sum_{i=1}^{n}(|y_i - \widehat{y_i}|) \tag{9}$$

While RMSE and MAE serve as the principal indicators of forecasting accuracy, the coefficient of determination (R^2) is a supplementary measure that assesses how well the model explains the data's variance. Its value lies between 0 and 1, where values approaching 1 represent a stronger model fit. As illustrated in Equation 10, R^2 indicates the model's ability to capture demand patterns, encompassing long-term trends as well as cyclical variations, thereby offering insight into the extent to which the model captures such dynamics [50]. The formula for R^2 is presented in Equation 10 [51].

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \widehat{y_i})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{10}$$

2.3.7. Overfitting and underfitting

During training, the model was continuously monitored to detect signs of underfitting or overfitting. When such issues were identified, the model was reconfigured accordingly. Underfitting occurs when the model performs poorly on both training and unseen data, typically due to insufficient training, overly strong regularization, or a lack of relevant predictive features [52]. In contrast, overfitting occurs when the model is overly flexible in fitting the training data, leading to poor generalization to unseen data. This is often caused by limited data or by high variability not adequately captured by the available features, as frequently observed in ecological datasets [53][54].

3. RESULTS AND DISCUSSION

This section presents the experimental results based on the methodology described earlier. The results are provided to demonstrate the model's performance, evaluate the effectiveness of the proposed approach, and compare it with relevant benchmarks. These findings form the basis for discussing their implications and significance.

3.1. Results

In this study, the LSTM architecture was employed, as it is computationally lighter than Transformer-based models. This characteristic enables efficient training without the need for large-scale computational resources. Therefore, the hardware and software specifications used in this research are presented in Table 1, serving as a reference for assessing computational requirements and the potential for replicating the study.

Table 1. Specification Compute Unit

Component	Specification
GPU	RTX 2060 12 GB
CPU	Ryzen 5 5600
RAM	24 GB
Storage	SSD NVME 256 GB
Framework	PyTorch 2.8.0+cu128

Table 2. k-Fold Cross-Validation

k-Fold						Average Score			Training Time (Second)
Inputs	Optimizer	Stock	Layers	Hidden Size	Epoch	RMSE	MAE	R^2	
Close	Adam Optimizer	LW	1	64	1000	0.0520	0.0402	0.6568	25.8
			2	64	1000	0.0620	0.0490	0.5360	31
			1	128	1000	0.0541	0.0429	0.6351	26.3
			2	128	1000	0.0608	0.0480	0.5276	39.1
		LKQ	1	64	1000	0.1694	0.1462	-1.6200	26
			2	64	1000	0.1676	0.1435	-1.3401	30.3
			1	128	1000	0.1296	0.1042	0.2005	26
			2	128	1000	0.1986	0.1683	-2.0171	40.4
		IPG	1	64	1000	0.0876	0.0704	0.6999	25.9
			2	64	1000	0.0932	0.0751	0.6245	30.2
			1	128	1000	0.0880	0.0704	0.7024	26.3
			2	128	1000	0.0954	0.0770	0.6105	40
		MGM	1	64	1000	0.1138	0.0896	0.5838	25.8
			2	64	1000	0.1188	0.0933	0.5170	31.6
			1	128	1000	0.1089	0.0844	0.6430	27.5
			2	128	1000	0.1199	0.0945	0.5161	39.8
		RL	1	64	1000	0.1276	0.0997	0.5676	26.3
			2	64	1000	0.1453	0.1135	0.4146	30.4
			1	128	1000	0.1303	0.0998	0.5355	27
			2	128	1000	0.1685	0.1333	0.1844	40.2
		CAG	1	64	1000	0.0980	0.0735	0.5403	25.6
			2	64	1000	0.1068	0.0824	0.4019	30.3
			1	128	1000	0.0972	0.0738	0.5403	25.4
			2	128	1000	0.1152	0.0909	0.2638	40.8
Close, MA 200, Stochastic	Adam Optimizer	LW	1	64	1000	0.0972	0.0683	0.9018	25.5
			2	64	1000	0.1121	0.0779	0.8717	29.7
			1	128	1000	0.1034	0.0725	0.8887	26
			2	128	1000	0.1346	0.0924	0.8096	40.1
		LKQ	1	64	1000	0.3379	0.2367	0.0593	26.1
			2	64	1000	0.2619	0.1912	0.4771	30.1
			1	128	1000	0.3089	0.2209	0.2048	27.7
			2	128	1000	0.2656	0.2059	0.4275	40.5
		IPG	1	64	1000	0.2282	0.1681	0.7586	26.4
			2	64	1000	0.2444	0.1848	0.749	29.8
			1	128	1000	0.2092	0.1499	0.8042	25.9
			2	128	1000	0.2749	0.1980	0.6490	39
		MGM	1	64	1000	0.2817	0.2022	0.5974	25.4
			2	64	1000	0.2642	0.1963	0.6318	30
			1	128	1000	0.2423	0.1802	0.7483	25.8
			2	128	1000	0.2873	0.2142	0.5989	40
		RL	1	64	1000	0.2811	0.2035	0.5902	25.2
			2	64	1000	0.2673	0.1963	0.6521	29.9
			1	128	1000	0.2892	0.2116	0.5938	25.7
			2	128	1000	0.2876	0.2070	0.6075	40
		CAG	1	64	1000	0.1856	0.1331	0.7612	25.5
			2	64	1000	0.1950	0.1421	0.7604	29.8
			1	128	1000	0.1683	0.1206	0.7895	26.1
			2	128	1000	0.2094	0.1535	0.7179	39.2

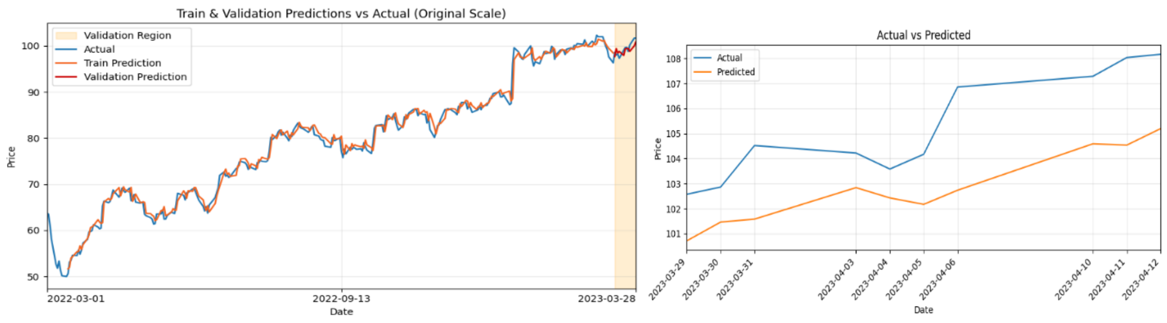
Subsequently, Table 2 presents the experimental results obtained using the k-Fold Cross-Validation with different configurations of layer numbers and hidden unit sizes. The evaluation was conducted across multiple stocks to assess the model's performance consistency. The key parameters observed include Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and the coefficient of determination (R^2). In addition, training time was also recorded to assess the computational efficiency of each configuration.

Table 3. Split-Validation from Previous Studies [18]

Split-Validation		Average RMSE							
Inputs	Optimizer	Layers LSTM	Epoch	LW	LKQ	IPG	MGM	RL	CAG
Close	Without Optimizer	1	10	1.18	-	-	-	-	-
Close	Adam	1	10	-	0.70	-	-	-	-

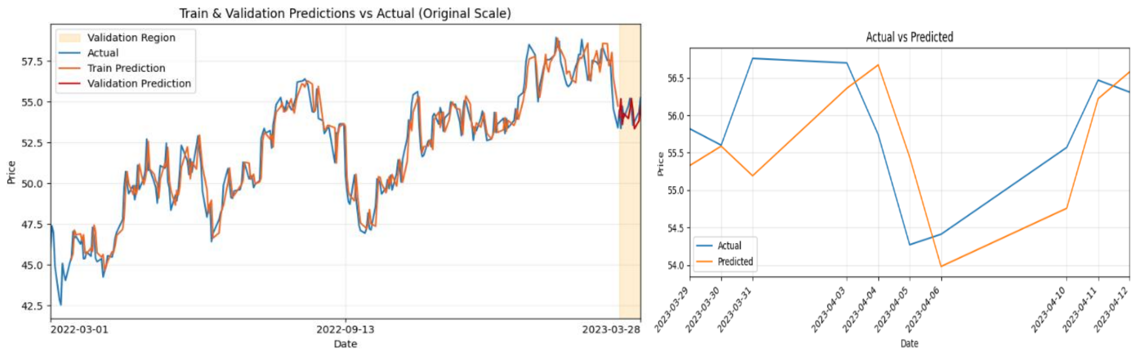
Split-Validation		Average RMSE							
Inputs	Optimizer	Layers LSTM	Epoch	LW	LKQ	IPG	MGM	RL	CAG
Close, MA 200, Stochastic	Adam	1	100	-	-	0.45	-	-	-
Close	Without Optimizer	1	100	-	-	-	0.841	-	-
Close	Without Optimizer	1	100	-	-	-	-	1.544	-
Close, MA 200, Stochastic	Without Optimizer	2	20	-	-	-	-	-	0.130

After the evaluation results are presented in tabular form, the analysis continues by illustrating the model's training and prediction process. This visualization illustrates how the model adapts to the data and the extent to which its predictions align with the actual price patterns, as shown in Figures 3 through 8.



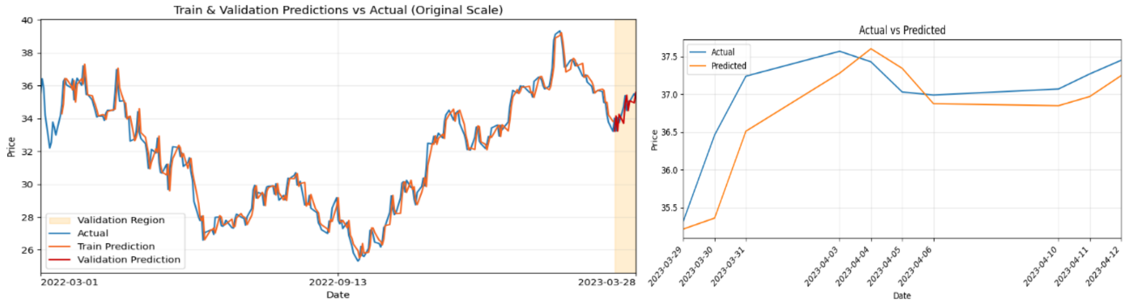
(a) (b)

Figure 3. LW (a) Train, (b) Prediction



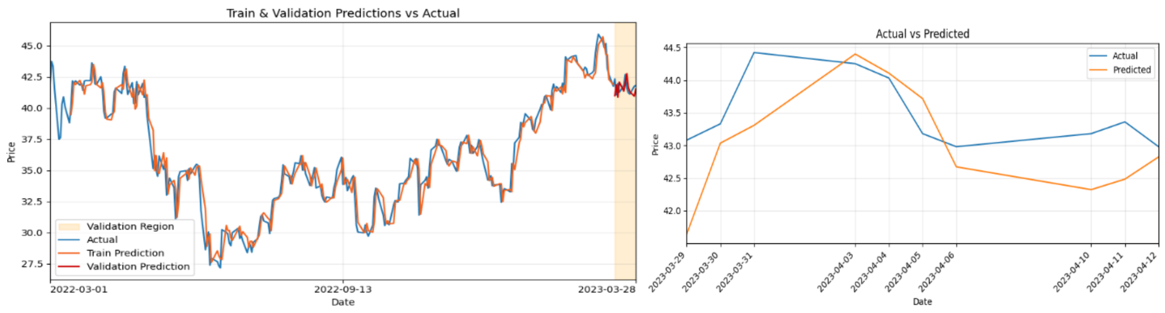
(a) (b)

Figure 4. LKQ (a) Train, (b) Prediction



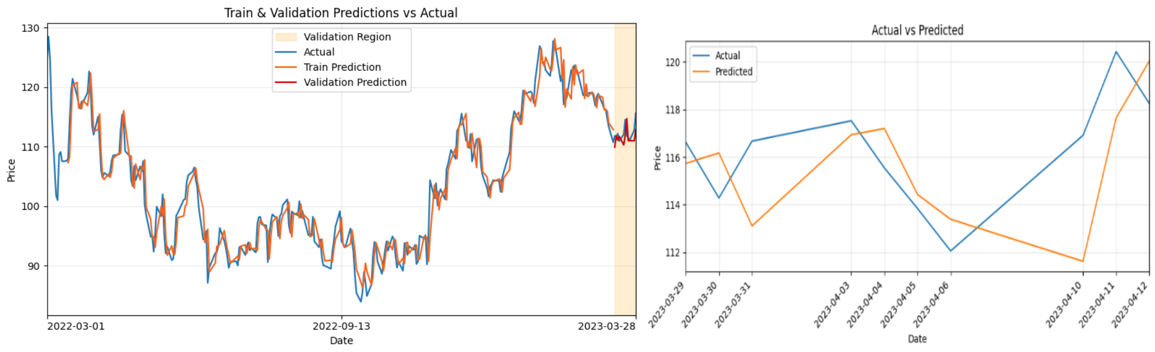
(a) (b)

Figure 5. IPG (a) Train, (b) Prediction



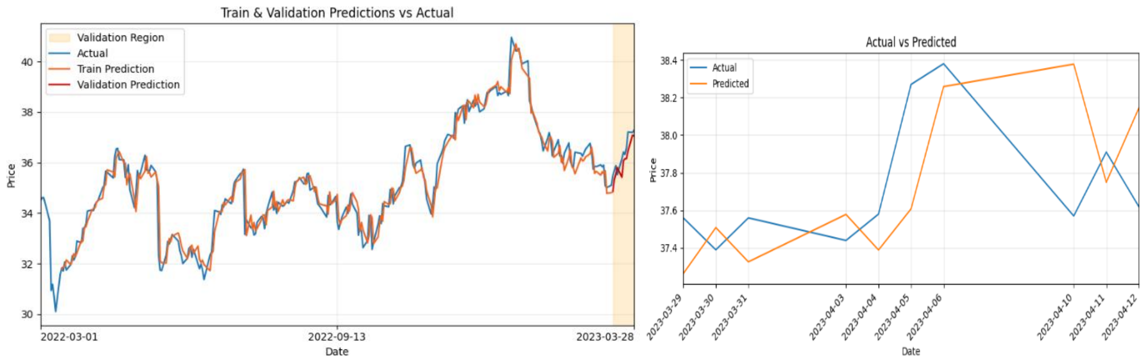
(a) (b)

Figure 6. MGM (a) Train, (b) Prediction



(a) (b)

Figure 7. RL (a) Train, (b) Prediction



(a) (b)

Figure 8. CAG (a) Train, (b) Prediction

3.1.1. Comparative Performance Analysis k-Fold vs. Split Validation

Based on the results presented in Table 2 and Table 3, the best Root Mean Square Error (RMSE) values for each approach, k-Fold Cross-Validation and Split Validation, can be observed. The overall best results from both methods are summarized in Table 4, and a visual comparison is provided in Figure 9, which clearly illustrates the performance differences across the various input configurations and validation techniques.

Table 4. Best RMSE Results for Each Input

Stock	k-Fold Close RMSE	k-Fold (Close+MA200+Stochatic) RMSE	Split Validation RMSE
LW	0.0520	0.0972	1.180
LKQ	0.1296	0.2619	0.700
IPG	0.0876	0.2092	0.450
MGM	0.1089	0.2423	0.841
RL	0.1276	0.2673	1.544
CAG	0.0972	0.1683	0.130

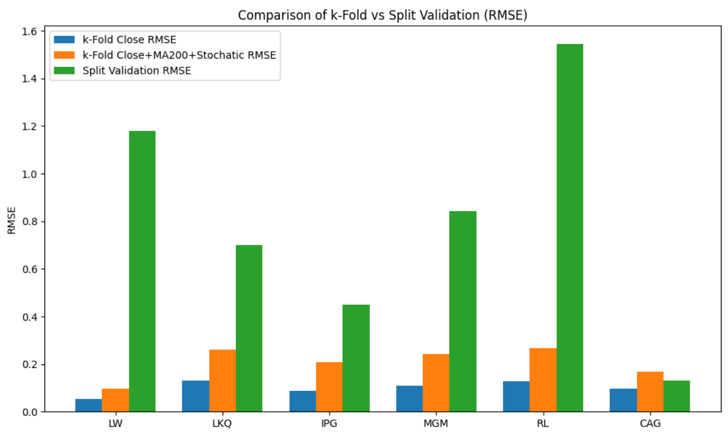


Figure 9. Comparison of RMSE between k-Fold and Split Validation

Based on the results presented in Table 4 and Figure 9, the k-Fold validation method consistently yields significantly lower RMSE values than Split Validation. This indicates that the k-Fold approach provides more stable and accurate error estimation, because it iteratively utilizes the entire dataset during validation. For instance, in the case of LW stock, the RMSE obtained from k-Fold is only 0.0520, whereas Split Validation reaches 1.180. A similar pattern is also evident in RL stock, where the difference is even more pronounced, with k-Fold achieving an RMSE of 0.1276 compared to 1.544 with split Validation. The only exception is CAG stock, where Split Validation slightly outperforms k-Fold with additional indicators (0.1300 vs. 0.1683).

These findings are further supported by a two-tailed paired t-test conducted in Excel. As shown in Table 5, the comparison between k-Fold (Close) and Split Validation yields a *p*-value of 0.0185, which is below the 0.05 significance threshold. This *p*-value indicates that the probability of obtaining an RMSE difference of this magnitude (or more extreme) is only about 1.85%, assuming the null hypothesis of no difference between the two methods is true. In other words, the result provides sufficient evidence to reject the null hypothesis and conclude that a statistically significant difference exists.

Table 5. Paired t-Test Result k-Fold (Close) vs Split

Stock	k-Fold Close RMSE	Split Validation RMSE
LW	0.0520	1.180
LKQ	0.1296	0.700
IPG	0.0876	0.450
MGM	0.1089	0.841
RL	0.1276	1.544
CAG	0.0972	0.130
p		0.018592744

Furthermore, Table 6 presents the results of the paired *t*-test between k-Fold (Close+MA200+Stochastic) and Split Validation, which yields a *p*-value of 0.0324, below the 0.05 threshold. This suggests that the probability of the observed difference arising by chance is approximately 3.24%, thus supporting the conclusion that a statistically significant difference exists.

Table 6. Paired t-Test Result k-Fold (Close+MA200+Stochastic) vs Split

Stock	k-Fold (Close+MA200+Stochatic) RMSE	Split Validation RMSE
LW	0.0972	1.180
LKQ	0.2619	0.700
IPG	0.2092	0.450
MGM	0.2423	0.841
RL	0.2673	1.544
CAG	0.1683	0.130
p		0.032478408

Overall, this analysis confirms that k-Fold Cross-Validation is strongly recommended for stock time-series forecasting, as it leverages the entire dataset iteratively, producing more representative error estimates than Split Validation, which relies on a single data partition. Nevertheless, it is essential to note that the inclusion of additional technical indicators does not always guarantee improved accuracy. In some cases, the use of MA200 and stochastic indicators even slightly increased the error compared to models that rely solely on closing price (Close) data.

3.1.2. Comparison between Close input and Close+MA200+Stochastic

Table 2 presents an interesting finding regarding the R² metric. In the k-Fold scheme, including technical indicators yields substantially higher R² values; however, the RMSE and MAE are lower for the model that uses only Close as input. This outcome appears to contradict the initial expectation since additional features are generally expected to improve predictive accuracy.

This discrepancy can be clarified when the analysis objective is clearly defined from the outset. If the primary goal is to achieve the most accurate price prediction, then RMSE and MAE are the more relevant metrics. The experimental results indicate that the model with Close-only input consistently yields smaller error values, thereby outperforming in approximating actual prices. In other words, the inclusion of technical indicators does not enhance performance in the context of short-term predictive accuracy, and in some cases, it even increases absolute errors.

On the other hand, if the analysis aims to understand the dynamics or trends of price movements, technical indicators are useful. Although they do not always reduce RMSE, the combination of Close, MA200, and Stochastic significantly improves the R² value. For example, in the case of LW stock, the R² increased from 0.6568 to 0.9018. This suggests that the model with additional features is better able to explain price movements, even if numerical prediction accuracy does not necessarily improve.

This phenomenon may occur because the addition of technical analysis does not always yield new or truly relevant information about short-term price patterns. Instead, additional features may increase data complexity and noise, making it harder for the model to capture the direct relationship between the input variables and price. As a result, numerical accuracy metrics such as RMSE and MAE may fail to improve and can even deteriorate. Nevertheless, technical indicators still play an essential role in enriching the context of price movements, enabling the model to provide better explanations of market dynamics, as reflected in the improved R² values.

3.1.3. Evaluation of Training Efficiency

In addition to accuracy metrics, training time provides valuable insights into computational efficiency, which is a critical factor in real-world forecasting applications. As shown in Table 2, single-layer LSTM configurations required approximately 25-27 seconds on average, while adding a second layer increased training time to approximately 30-40 seconds. Although deeper architectures theoretically increase the model's representational capacity, the additional computational overhead did not consistently lead to improved accuracy, as in several cases, RMSE and MAE either remained comparable or worsened. This trade-off underscores that more complex configurations are not always optimal for time-series forecasting.

A noteworthy practical finding from this study is that the LSTM architecture does not require high-end computational resources to achieve competitive performance. As indicated in Table 1, the experiments were conducted using mid-range hardware, an RTX 2060 GPU (12 GB VRAM), a Ryzen 5 5600 CPU, and 24 GB of RAM, yet training remained highly efficient. This suggests that accurate stock forecasting models can be trained and deployed on relatively modest setups without requiring specialized high-performance computing infrastructure. Such efficiency broadens the accessibility of LSTM-based forecasting methods for both academic research and industrial applications, particularly in contexts with limited hardware resources.

Overall, the results confirm that lightweight LSTM configurations strike an effective balance between prediction accuracy, training efficiency, and computational resource requirements. While deeper models introduce higher computational costs without consistent benefits, simpler configurations remain robust and practical, reinforcing LSTMs' role as a viable, resource-efficient alternative to computationally demanding architectures such as Transformers.

3.2. Discussion

The findings of this study offer important insights from both methodological and practical perspectives. First, the LSTM architecture demonstrated competitive performance in stock time-series forecasting, even when implemented on mid-range hardware. This confirms that developing accurate predictive models does not necessarily require high-end computing infrastructure, thereby broadening their applicability for researchers and practitioners in resource-constrained environments.

In terms of accuracy, the comparative analysis of k-Fold Cross-Validation and Split Validation shows that k-Fold consistently yields lower and more stable error estimates. This implies that k-Fold is more reliable and thus recommended for stock forecasting tasks, particularly when the dataset size is limited. This strengthens the argument that rigorous evaluation practices are essential to reduce the risk of overfitting and ensure better model generalization.

Furthermore, comparing Close-only input with Close+MA200+Stochastic shows that including technical indicators does not continually improve numerical accuracy, though it yields higher R² values. This has important implications for practical applications: a simpler model with close-only input is more suitable for short-term price forecasting, whereas models with additional indicators are better suited for explaining market dynamics more comprehensively. In other words, the choice of input configuration should align with the user's specific needs, whether the focus is on short-term quantitative forecasting or on capturing long-term market trends.

The practical implications of this research extend to several contexts. Retail and institutional investors may employ LSTM-based models as lightweight, efficient decision-support tools that can be deployed without costly infrastructure. Moreover, these findings are relevant to developers of algorithmic trading systems requiring fast, low-latency predictions. Since training efficiency can be achieved even on mid-range hardware, integrating such models into real-time trading systems is increasingly feasible.

3.2.1. Limitations

This study is subject to several limitations. First, the dataset is restricted to a small set of selected stocks, which may limit the generalizability of the findings to the broader market. Second, the technical indicators employed in this study are limited to MA200 and the Stochastic Oscillator, leaving out a wider range of potentially relevant indicators. Third, although k-Fold Cross-Validation has been shown to outperform Split Validation, it still carries a risk of data leakage if not applied carefully in time-series contexts. Therefore, these results should be viewed as a foundation for further research rather than a definitive statement on the effectiveness of LSTM models across all stock market scenarios.

4. CONCLUSION

This study has demonstrated the effectiveness of LSTM models for stock time-series forecasting by comparing different validation strategies, input configurations, and architectural complexities. The experimental results highlight three key findings. First, k-Fold Cross-Validation consistently produced more reliable, lower RMSE values than Split Validation, confirming its suitability for time-series forecasting tasks where data efficiency and robustness are crucial. Second, although it was initially expected that adding technical indicators (MA200 and the Stochastic Oscillator) would improve predictive accuracy, the results showed the opposite: models using only Close data achieved lower RMSE and MAE, whereas the inclusion of additional indicators primarily improved the R^2 metric. This surprising outcome suggests that while technical indicators may improve the model's explanatory power in capturing market dynamics, they do not necessarily translate into better short-term predictive accuracy. Third, from a computational perspective, the findings reaffirm that LSTM can deliver competitive performance with modest hardware, making it a practical choice for academic and industrial applications, particularly in resource-constrained environments.

Based on these results, several recommendations for future work can be made. First, further research should investigate the integration of a broader range of technical and macroeconomic features, including sentiment analysis from financial news or social media, to evaluate their contribution to predictive performance. Second, exploring hybrid architectures that combine LSTM with attention mechanisms or Transformer-based models to balance computational efficiency with predictive accuracy. Finally, testing the model across longer time horizons and diverse market conditions would strengthen the evidence regarding its robustness and generalizability.

REFERENCES

[1] H. Bhandari, B. Rimal, N. Pokhrel, R. Rimal, K. Dahal, & R. Khatri, "Predicting Stock Market Index using LSTM", *Machine Learning with Applications*, vol. 9, pp. 100320, 2022. https://doi.org/10.1016/j.mlwa.2022.100320

[2] J. Shah, D. Vaidya, & M. Shah, "A Comprehensive Review on Multiple Hybrid Deep Learning Approaches for Stock Prediction", *Intelligent Systems with Applications*, vol. 16, pp. 200111, 2022. https://doi.org/10.1016/j.iswa.2022.200111

[3] F. Furizal, A. Ritonga, A. Ma'arif, & I. Suwarno "Stock Price Forecasting with Multivariate Time Series Long Short-Term Memory: A Deep Learning Approach", *Journal of Robotics and Control*, vol. 5, no. 5, pp. 1322-1335, 2024. https://doi.org/10.18196/jrc.v5i5.22460

[4] R. Zhang, "LSTM-based Stock Prediction Modeling and Analysis", *Advances in Economics, Business and Management Research*, 2022. https://doi.org/10.2991/aebmr.k.220307.414

[5] H. Qian, "Stock Predicting based on LSTM and ARIMA", *Advances in Economics, Business and Management Research*, pp. 485-490, 2022. https://doi.org/10.2991/978-94-6463-036-7_72

[6] C. Jiang, Y. Tsai, Z. Shao, S. Lee, C. Hsueh, & K. Huang, "Hybrid Crow Search Algorithm–LSTM System for Enhanced Stock Price Forecasting", *Applied Sciences*, vol. 14, no. 23, pp. 11380, 2024. https://doi.org/10.3390/app142311380

[7] K. Liagkouras and K. Metaxiotis, "A Hybrid Long Short-Term Memory with a Sentiment Analysis System for Stock Market Forecasting", *Electronics*, vol. 14, no. 14, pp. 2753, 2025. https://doi.org/10.3390/electronics14142753

[8] M. Ali, D. Khan, H. Alshanbari, & A. El-Bagoury, "Prediction of Complex Stock Market Data Using an Improved Hybrid EMD-LSTM Model", *Applied Sciences*, vol. 13, no. 3, pp. 1429, 2023. https://doi.org/10.3390/app13031429

[9] P. Ye, H. Zhang, & X. Zhou, "CNN-CBAM-LSTM: Enhancing Stock Return Prediction Through Long and Short Information Mining in Stock Prediction", *Mathematics*, vol. 12, no. 23, pp. 3738, 2024. https://doi.org/10.3390/math12233738

[10] F. Fozap, "Hybrid Machine Learning Models for Long-Term Stock Market Forecasting: Integrating Technical Indicators", *Journal of Risk and Financial Management*, vol. 18, no. 4, pp. 201, 2025. https://doi.org/10.3390/jrfm18040201

[11] Q. Li, N. Kamaruddin, S. Yuhaniz, & H. Al-Jaifi, "Forecasting Stock Prices Changes using Long-Short Term Memory Neural Network with Symbolic Genetic Programming", *Scientific Reports*, vol. 14, no. 1, 2024. https://doi.org/10.1038/s41598-023-50783-0

[12] G. Sonkavde, D. Dharrao, A. Bongale, S. Deokate, D. Doreswamy, & S. Bhat, "Forecasting Stock Market Prices Using Machine Learning and Deep Learning Models: A Systematic Review, Performance Analysis and Discussion of Implications", *International Journal of Financial Studies*, vol. 11, no. 3, pp. 94, 2023. https://doi.org/10.3390/ijfs11030094

[13] A. Kid, M. Ahmed, A. Alkali, J. Usman, & A. Hashim, "Real-Time Energy Demand Forecasting and Adaptive Demand Response Optimization for IoT-Enabled Smart Grids", *Vokasi Unesa Bulletin of Engineering, Technology and Applied Science*, vol. 2, no. 2, pp. 366-375, 2025. https://doi.org/10.26740/vubeta.v2i2.36818

[14] B. Gülmez, "A Hybrid Approach for Stock Market Price Forecasting Using Long Short-Term Memory and Seahorse Optimization Algorithm", *Annals of Data Science*, 2025. https://doi.org/10.1007/s40745-025-00609-9

[15] E. Radfar, "Stock Market Trend Prediction using Deep Neural Network via Chart Analysis: A Practical Method or A Myth?", *Humanities and Social Sciences Communications*, vol. 12, no. 1, 2025. https://doi.org/10.1057/s41599-025-04761-8

[16] V. Varadharajan, N. Smith, D. Kalla, G. Kumar, F. Samaah, & K. Polimetla, "Stock Closing Price and Trend Prediction with LSTM-RNN", *Journal of Artificial Intelligence and Big Data*, vol. 4, no. 1, pp. 1-13, 2024. https://doi.org/10.31586/jaibd.2024.877

[17] V. Chang, Q. Xu, A. Chidozie, & H. Wang, "Predicting Economic Trends and Stock Market Prices with Deep Learning and Advanced Machine Learning Techniques", *Electronics*, vol. 13, no. 17, pp. 3396, 2024. https://doi.org/10.3390/electronics13173396

[18] I. Agung, T. Arifin, E. Junianto, M. Rabbani, & A. Mayangsari, "Stock's Selection and Trend Prediction using Technical Analysis and Artificial Neural Network", *International Journal of Advances in Applied Sciences*, vol. 14, no. 1, pp. 151, 2025. https://doi.org/10.11591/ijaas.v14.i1.pp151-163

[19] T. Ogundunmade, A. Adepoju, & A. Allam, "Stock Price Forecasting: Machine Learning Models with K-fold and Repeated Cross Validation Approaches", *Modern Economy and Management*, vol. 1, pp. 1, 2022. https://doi.org/10.53964/mem.2022001

[20] S. Zaheer, N. Anjum, S. Hussain, A. Algarni, J. Iqbal, S. Bourouis et al., "A Multi Parameter Forecasting for Stock Time Series Data Using LSTM and Deep Learning Model", *Mathematics*, vol. 11, no. 3, pp. 590, 2023. https://doi.org/10.3390/math11030590

[21] X. Kong, Z. Chen, W. Liu, K. Ning, L. Zhang, S. Marier et al., "Deep Learning for Time Series Forecasting: A Survey", *International Journal of Machine Learning and Cybernetics*, vol. 16, no. 7-8, pp. 5079-5112, 2025. https://doi.org/10.1007/s13042-025-02560-w

[22] D. El-Shahat, A. Tolba, M. Abouhawwash, & M. Abdel-Basset, "Machine Learning and Deep Learning Models based Grid Search Cross Validation for Short-Term Solar Irradiance Forecasting", *Journal of Big Data*, vol. 11, no. 1, 2024. https://doi.org/10.1186/s40537-024-00991-w

[23] X. Song, L. Deng, H. Wang, Y. Zhang, Y. He, & W. Cao, "Deep Learning-based Time Series Forecasting", *Artificial Intelligence Review*, vol. 58, no. 1, 2024. https://doi.org/10.1007/s10462-024-10989-8

[24] M. Krichen and A. Mihoub, "Long Short-Term Memory Networks: A Comprehensive Survey", *Ai*, vol. 6, no. 9, pp. 215, 2025. https://doi.org/10.3390/ai6090215

[25] I. Malashin, В. Тынченко, A. Gantimurov, V. Nelyub, & А. Бородулин, "Applications of Long Short-Term Memory (LSTM) Networks in Polymeric Sciences: A Review", *Polymers*, vol. 16, no. 18, pp. 2607, 2024. https://doi.org/10.3390/polym16182607

[26] C. Ubal, G. Di-Giorgi, J. Contreras-Reyes, & R. Salas, "Predicting the Long-Term Dependencies in Time Series Using Recurrent Artificial Neural Networks", *Machine Learning and Knowledge Extraction*, vol. 5, no. 4, pp. 1340-1358, 2023. https://doi.org/10.3390/make5040068

[27] V. Naghashi, M. Boukadoum, & A. Diallo, "Should We Reconsider RNNs for Time-Series Forecasting?", *Ai*, vol. 6, no. 5, pp. 90, 2025. https://doi.org/10.3390/ai6050090

[28] S. Mu, B. Liu, G. Jijian, C. Lien, & N. Nedjah, "Research on Stock Index Prediction Based on the Spatiotemporal Attention BiLSTM Model", *Mathematics*, vol. 12, no. 18, pp. 2812, 2024. https://doi.org/10.3390/math12182812

[29] S. Shi, F. Li, & W. Li, "A Hybrid Long Short-Term Memory-Graph Convolutional Network Model for Enhanced Stock Return Prediction: Integrating Temporal and Spatial Dependencies", *Mathematics*, vol. 13, no. 7, pp. 1142, 2025. https://doi.org/10.3390/math13071142

[30] M. Kabir, D. Bhadra, M. Ridoy, & M. Milanova, "LSTM–Transformer-Based Robust Hybrid Deep Learning Model for Financial Time Series Forecasting", *Sci*, vol. 7, no. 1, pp. 7, 2025. https://doi.org/10.3390/sci7010007

[31] S. Sang and L. Li, "A Novel Variant of LSTM Stock Prediction Method Incorporating Attention Mechanism", *Mathematics*, vol. 12, no. 7, pp. 945, 2024. https://doi.org/10.3390/math12070945

[32] O. Chamorro-Atalaya, J. Arévalo-Tuesta, D. Balarezo-Mares, A. Pacheco, O. Mendoza-León, M. Silvestre et al., "K-Fold Cross-Validation through Identification of the Opinion Classification Algorithm for the Satisfaction of University Students", *International Journal of Online and Biomedical Engineering (iJOE)*, vol. 19, no. 11, 2023. https://doi.org/10.3991/ijoe.v19i11.39887

[33] T. Leinonen, D. Wong, A. Vasankari, A. Wahab, R. Nadarajah, M. Kaisti et al., "Empirical Investigation of Multi-Source Cross-Validation in Clinical ECG Classification", *Computers in Biology and Medicine*, vol. 183, pp. 109271, 2024. https://doi.org/10.1016/j.compbiomed.2024.109271

[34] D. Wilimitis and C. Walsh, "Practical Considerations and Applied Examples of Cross-Validation for Model Development and Evaluation in Health Care: Tutorial", *Jmir Ai*, vol. 2, pp. e49023, 2023. https://doi.org/10.2196/49023

[35] T. Bradshaw, Z. Huemann, J. Hu, & A. Rahmim, "A Guide to Cross-Validation for Artificial Intelligence in Medical Imaging", *Radiology: Artificial Intelligence*, vol. 5, no. 4, 2023. https://doi.org/10.1148/ryai.220232

[36] J. Allgaier and R. Pryss, "Cross-Validation Visualized: A Narrative Guide to Advanced Methods", *Machine Learning and Knowledge Extraction*, vol. 6, no. 2, pp. 1378-1388, 2024. https://doi.org/10.3390/make6020065

[37] A. Aziz, M. Yusoff, W. Yaacob, & Z. Mustaffa, "Repeated Time-Series Cross-Validation: A New Method to Improved COVID-19 Forecast Accuracy in Malaysia", *MethodsX*, vol. 13, pp. 103013, 2024. https://doi.org/10.1016/j.mex.2024.103013

[38] P. Ali, "Investigating the Impact of Min-Max Data Normalization on the Regression Performance of K-Nearest Neighbor with Different Similarity Measurements", *Aro-the Scientific Journal of Koya University*, vol. 10, no. 1, pp. 85-91, 2022. https://doi.org/10.14500/aro.10955

[39] X. Cao, I. Stojković, & Z. Obradović, "A Robust Data Scaling Algorithm to Improve Classification Accuracies in Biomedical Data", *BMC Bioinformatics*, vol. 17, no. 1, 2016. https://doi.org/10.1186/s12859-016-1236-x

[40] T. Rygh, C. Vaage, S. Westgaard, & P. Lange, "Inflation Forecasting: LSTM Networks vs. Traditional Models for Accurate Predictions", *Journal of Risk and Financial Management*, vol. 18, no. 7, pp. 365, 2025. https://doi.org/10.3390/jrfm18070365

[41] V. Teodorescu and L. Brașoveanu, "Assessing the Validity of k-Fold Cross-Validation for Model Selection: Evidence from Bankruptcy Prediction Using Random Forest and XGBoost", *Computation*, vol. 13, no. 5, pp. 127, 2025. https://doi.org/10.3390/computation13050127

[42] K. Kinast and E. Fokoué, "An In-Depth Look at Rising Temperatures: Forecasting with Advanced Time Series Models in Major US Regions", *Forecasting*, vol. 6, no. 3, pp. 815-838, 2024. https://doi.org/10.3390/forecast6030041

[43] L. Alsmadi, G. Lei, & L. Li, "Forecasting Day-Ahead Electricity Demand in Australia Using a CNN-LSTM Model with an Attention Mechanism", *Applied Sciences*, vol. 15, no. 7, pp. 3829, 2025. https://doi.org/10.3390/app15073829

[44] T. Dinh, G. Thirunavukkarasu, M. Seyedmahmoudian, S. Mekhilef, & A. Stojcevski, "Predicting Commercial Building Energy Consumption Using a Multivariate Multilayered Long-Short Term Memory Time-Series Model", *Applied Sciences*, vol. 13, no. 13, pp. 7775, 2023. https://doi.org/10.3390/app13137775

[45] M. Diqi, I. Ordiyasa, & H. Hamzah, "Enhancing Stock Price Prediction Using Stacked Long Short-Term Memory", *IT Journal Research and Development*, vol. 8, no. 2, pp. 164-174, 2024. https://doi.org/10.25299/itjrd.2023.13486

[46] Z. Wang, "Stock Price Prediction using LSTM Neural Networks: Techniques and Applications", *Applied and Computational Engineering*, vol. 86, no. 1, pp. 294-300, 2024. https://doi.org/10.54254/2755-2721/86/20241605

[47] F. Peng, X. Ji, L. Zhang, J. Wang, K. Zhang, & W. Wu, "Interpretable Mixture of Experts for Decomposition Network on Server Performance Metrics Forecasting", *Electronics*, vol. 13, no. 20, pp. 4116, 2024. https://doi.org/10.3390/electronics13204116

[48] C. Lei, H. Zhang, W. Zhi-gang, & Q. Miao, "Deep Learning for Demand Forecasting: A Framework Incorporating Variational Mode Decomposition and Attention Mechanism", *Processes*, vol. 13, no. 2, pp. 594, 2025. https://doi.org/10.3390/pr13020594

[49] M. Abumohsen, A. Owda, & M. Owda, "Electrical Load Forecasting Using LSTM, GRU, and RNN Algorithms", *Energies*, vol. 16, no. 5, pp. 2283, 2023. https://doi.org/10.3390/en16052283

[50] C. Lei, H. Zhang, W. Zhi-gang, & Q. Miao, "Multi-Model Fusion Demand Forecasting Framework Based on Attention Mechanism", *Processes*, vol. 12, no. 11, pp. 2612, 2024. https://doi.org/10.3390/pr12112612

[51] J. Olaniyan, D. Olaniyan, I. Obagbuwa, B. Esiefarienrhe, A. Adebiyi, & O. Bernard, "Intelligent Financial Forecasting with Granger Causality and Correlation Analysis Using Bayesian Optimization and Long Short-Term Memory", *Electronics*, vol. 13, no. 22, pp. 4408, 2024. https://doi.org/10.3390/electronics13224408

[52] C. Xu, P. Coen-Pirani, & X. Jiang, "Empirical Study of Overfitting in Deep Learning for Predicting Breast Cancer Metastasis", *Cancers*, vol. 15, no. 7, pp. 1969, 2023. https://doi.org/10.3390/cancers15071969

[53] A. Rezaeezade and L. Batina, "Regularizers to the Rescue: Fighting Overfitting in Deep Learning-based Side-channel Analysis", *Journal of Cryptographic Engineering*, vol. 14, no. 4, pp. 609-629, 2024. https://doi.org/10.1007/s13389-024-00361-5

[54] E. Chollet, A. Scheidegger, J. Wydler, & N. Schuwirth, "A Comparison of Machine Learning and Statistical Species Distribution Models: Quantifying Overfitting Supports Model Interpretation", *Ecological Modelling*, vol. 481, pp. 110353, 2023. https://doi.org/10.1016/j.ecolmodel.2023.110353

.

BIOGRAPHIES OF AUTHORS

Rifki Ainul Yaqin is a bachelor's student in the Informatics Study Program, Faculty of Engineering at Adhirajasa Reswara Sanjaya (ARS) University, and works as a research assistant at ARS Digital Research and Innovation (ADRI). Previously, participated in research focused on two studies "*Penerapan Arsitektur Monolitik Pada Aplikasi Jasa Service Online Tekku Berbasis Web*" and "*Klasifikasi Jenis Jerawat Secara Otomatis Dengan Convolutional Neural Network Menggunakan Arsitektur Resnet-50*". His research interests include machine learning, deep learning, and large language models (LLMs). He can be contacted at email: rifkiainul17@gmail.com.

Muhammad Iqbal Anshori is a bachelor's student in the Informatics Study Program, Faculty of Engineering at Adhirajasa Reswara Sanjaya (ARS) University. Previously, participated in research focused on two studies *"Penerapan Arsitektur Monolitik Pada Aplikasi Jasa Service Online Tekku Berbasis Web"* and *"Klasifikasi Jenis Jerawat Secara Otomatis Dengan Convolutional Neural Network Menggunakan Arsitektur Resnet-50"*. His interests include software development focusing on backend development, microservices architecture, and system security, as well as artificial intelligence covering machine learning, deep learning, and large language models (LLMs). He can be contacted at email: iqbalanshr@gmail.com.

Reddis Angel is a bachelor's student in the Informatics Study Program, Faculty of Engineering at Adhirajasa Reswara Sanjaya (ARS) University. She has published papers on Google Scholar titled *"Pengembangan Platform E-Commerce UMKM Berbasis Laravel dengan Blackbox Testing dan Metode Waterfall"* and *"Penerapan Algoritma Backtracking Berbasis BFS dengan Pendekatan Heuristik dalam Permainan Hangman"*. Her research interest is in data science, and she is also interested in web programming. She can be contacted at email: ddisangel@gmail.com.

Ignatius Wiseto Prasetyo Agung He is now dedicated his time in the ARS (Adhirajasa Reswara Sanjaya) University Bandung, Indonesia, as a lecturer and Vice Rector for Collaboration & Innovation, since October 2019. In Telkom Indonesia, he worked since 1988 in various divisions e.g satellite development, network operation, R&D, and Digital Business. He received the Sarjana (Bachelor Degree) in Telecommunication from Institut Teknologi Bandung, Indonesia in 1987. He was also graduated from University of Surrey, UK and received the MSc in Telematics (1994) and PhD in Multimedia Communication (2002). He was also in charge in several professional forums, for instance the Asia Pacific Telecommunity Wireless Forum (AWF) as Convergence Working Group Chairman (2008- 2011); in ITU-D as Vice Rapporteur (2007-2009); as Chairman (2020, 2021) and Vice Chair (2018-2019) of IEEE Communications Society Indonesia Chapter; and as Senior Member of IEEE. He can be contacted at email: wiseto.agung@ars.ac.id.

Toni Arifin He is a member of the Faculty of Engineering, majoring in Informatics Engineering, Adhirajasa Reswara Sanjaya (ARS) University, and researcher ARS Digital Research & Innovation (ADRI). He received his Bachelor's Degree in Informatics Engineering from Bina Sarana Informatika University in 2013 and graduated from the computer science master's program at Nusa Mandiri University Jakarta in 2015. He has authored or coauthored more than 68 publications: 4 proceedings and 66 journals, with 12 H-index and more than 528 citations. Research interests include machine learning, image processing and deep learning. He can be contacted at email: toni.arifin@ars.ac.id

Erfian Junianto He is a member of the Faculty of Engineering, majoring in Informatics Engineering, at Adhirajasa Reswara Sanjaya (ARS) University, and a researcher at ARS Digital Research & Innovation (ADRI). He graduated from the computer science master's program at Nusa Mandiri University Jakarta in 2014. He has authored or co-authored more than 38 publications, including 2 proceedings and 36 journals, with an H-index of 10 and more than 450 citations. His research interests include text mining, artificial intelligence, and classification. He can be contacted at email: erfian.ejn@ars.ac.id