UJM 13 (2) 2024
UNNES Journal of Mathematics
http://journal.unnes.ac.id/sju/index.php/ujm

Bayesian Optimization for Stock Price Prediction Using LSTM, GRU, and Hybrid LSTM-GRU, and Hybrid GRU-LSTM

Mira Dwi Utami [*], Iqbal Kharisudin

Department of Mathematics, Faculty of Mathematics and Natural Sciences,
Universitas Negeri Semarang, Semarang, 50229, Indonesia

Article Info

Article History:
Recived 07-02-2025
Accepted …..
Published …..

Keywords:
Bayesian Optimization, Surrogate Models, Deep Learning, Stock Price Prediction

Abstract

Stocks have high price fluctuations, which include high risks and high potential returns for investors. This high potential return has attracted significant interest from investors. This study proposes the use of Bayesian optimization methods with Gaussian Process (GP), Random Forest (RF), Extra Trees (ET), and Gradient Boosted Regression Trees (GBRT) surrogate models to enhance the accuracy of stock price predictions using Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and hybrid models (LSTM-GRU and GRU-LSTM). This study tests the effectiveness of various combinations of hyperparameters optimized using the Bayesian optimization method. The model optimized with the Bayesian approach and the GP surrogate model demonstrates superior results compared to the others. Evaluation is conducted using Mean Square Error (MSE), Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), and R-squared (R^2) metrics. The results indicate that Bayesian optimization with the GP surrogate model for the GRU-LSTM hybrid model outperforms all other methods in terms of MSE, RMSE, MAE, MAPE, and R^2. These findings provide significant contributions to parameter selection for stock price prediction and demonstrate the great potential of using Bayesian optimization methods to improve the accuracy of prediction models.

How to cite:

Utami, M.D., & Kharisudin, I. 2024. Bayesian Optimization for Stock Price Prediction Using LSTM, GRU, Hybrid LSTM-GRU, and Hybrid GRU-LSTM. *UNNES Journal of Mathematics*. 13 (2): 9-19

*Correspondence Address
E-mail: miradwiutami02@gmail.com

p-ISSN 2252-6943
e-ISSN 2460-5859

1. Introduction

Capital market investors in Indonesia have experienced a consistent increase in recent years. The number of Indonesian capital market investors registered with the Kustodian Sentral Efek Indonesia (KSEI) based on the Single Investor Identification (SID) has reached 11.72 million as of September 2023. According to KSEI data from the same month, the majority of stock market investors in Indonesia are millennials and Generation Z, with ages under 30 and ages 31-40, comprising more than 80%. This phenomenon is driven by increasing awareness of the benefits of long-term investment in the capital market and the growing accessibility of information through online investment applications and platforms. Many of these investors are interested in capital market instruments such as stocks. Notably, the number of stock investors has increased by 13.27% in the last 10 months.

One of the most popular stock sectors is the banking sector. According to the Financial Services Authority (OJK), the banking industry in Indonesia has shown significant growth over the past five years. The banking sector plays a crucial role in improving the investment climate and supporting the growth of the capital market in Indonesia. Banking stocks, which constitute 33% of the stocks listed on the Indonesia Stock Exchange (IDX), dominate the capitalization of the capital market in Indonesia. These stocks also dominate among liquid stocks and large market capitalizations, contributing more than 45% to the market capitalization of the IDX80, LQ45, and IDX30 indices. One notable company included in these indices is PT Bank Central Asia Tbk (BCA).

PT Bank Central Asia Tbk is one of the largest banks in Indonesia. With the stock code BBCA, the bank's performance is among the best on the IDX. Its solid reputation and strong financial stability make BBCA shares a relatively safe choice for investors. Consistent growth year over year makes BBCA shares attractive for long-term investment. BBCA shares also have high liquidity, allowing for easy transactions, and offer attractive dividends that provide additional passive income for shareholders. The ease of access through online trading platforms further facilitates investment in Bank BCA shares. However, investors need to conduct thorough research before making investment decisions.

Stock price movements are influenced by various factors such as economic conditions, government policies, interest rate changes, and investor sentiment (Zhao *et al*., 2023), making the stock market highly volatile. Therefore, accurately predicting future stock prices is crucial for making effective investment decisions. Additionally, predicting stock prices can help investors reduce risks and increase returns in the stock investment process. With technological advances, deep learning techniques are increasingly used for time series prediction, including stock prices. Deep learning methods generally outperform traditional machine learning methods in stock price prediction due to their ability to model non-linear relationships and produce more accurate predictions (Gur, 2024). Various deep learning methods, such as Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), and Gated Recurrent Units (GRU), have been designed to handle time series data and have shown superior performance when previous events are important for predicting future events (Song & Choi, 2023).

Previous studies on stock price prediction, such as (Kervanci *et al*., 2024), have used hyperparameter optimization methods like Bayesian Optimization, Random Search, and Grid Search with LSTM, GRU, and hybrid LSTM-GRU models to predict Bitcoin prices. Their study found that Bayesian optimization with a hybrid LSTM-GRU model outperformed all other methods. Further research by Trivedi & Patel (2022) used GRU, LSTM, and hybrid GRU-LSTM models to predict stock prices and found that the hybrid GRU-LSTM model provided more accurate predictions than other methods.

Current trends in model development involve improving predictive accuracy by using various optimal algorithms or combining the advantages of different algorithms through hybrid methods. However, hybrid models have higher complexity than single models and require consideration of numerous parameters. Model performance can vary depending on parameter optimization, as overall performance can decrease if the parameters used are not optimal (Baek, 2023). This study proposes a hybrid model of LSTM and GRU with parameter optimization applied for stock price prediction. The model combines the strengths of LSTM, which can handle long-term dependencies with complex architectures, and GRU, which is simpler, more efficient, and can also handle long-term dependencies with fewer computing resources. For parameter optimization, the Bayesian Optimization (BO) technique is applied. BO is effective in optimizing parameters. In summary, this study proposes a hybrid model that integrates deep learning and BO algorithms to find a suitable model for predicting stock closing prices.

2. Method

2.1. Dataset

The dataset used in this study was downloaded from Yahoo Finance (https://finance.yahoo.com/quote/BBCA.JK/history/). It includes stock price information for PT Bank Central Asia Tbk (BBCA.JK) from May 2014 to May 2024. This data encompasses various trading details such as Open, High, Low, Close, Volume, and Adj Close. The dataset is time series data. In this research, optimization methods were applied with cross-validation = 3, a training size of 0.8, and each method was executed for 100 epochs.

2.2. Preprocessing data

Before inputting raw data into the model, data preprocessing needs to be done. First, normalize the data using Scikit-learn's MinMaxScaler tool, as follows (Zhang *et al*., 2023):

$$x = \frac{x - x_{max}}{x_{max} - x_{min}}, \tag{1}$$

where x is the input feature of the stock price data, and x_{max} and x_{min} are the maximum and minimum values of each input feature, respectively.

Next, the data is divided into a training set and a testing set. In this study, the data is divided into 80% for training and 20% for testing. For each stock price data point, the Open, High, Low, Volume, and Adj Close features are used as input to train the model, and the Close feature is used as output to predict one step and multiple steps ahead. To create input and output pairs, the previously normalized data is segmented using a sliding window technique. A fixed window size of the time series data is selected as input, and a fixed number of subsequent observations are selected as output. This process is repeated for the entire dataset by sliding the window in one-time-step intervals to obtain the next input and output. We train the proposed model on m consecutive past feature data. The input at time t is denoted by

$$X_t = (x_t^O, x_t^H, x_t^L, x_t^V, x_t^{Ac}) \in \mathbb{R}^{m\times 6}, \tag{2}$$

where for each $k \in \{O, H, L, V, Ac\}$,

x^O, x^H, x^L, x^V, dan x^{Ac} are the Open, High, Low, Volume, and Adj Close from time $t-m+1$ to time t, respectively. The input X_t is fed sequentially into the proposed model to predict the following n daily closing prices of the stock price, with the output denoted by

$$y_{t+1}^C = (y_{t+1}, y_{t+2}, \dots, y_{t+n})^T \in \mathbb{R}^m. \tag{3}$$

2.3. Model architectures

2.3.1. LSTM

LSTM, proposed by (Hochreiter & Schmidhuber, 1997), is a variant of RNN designed to address the vanishing or exploding gradient problem by incorporating a cell state in addition to the hidden state of the RNN (Song & Choi, 2023). LSTM architecture comprises three gates: the input gate, output gate, and forget gate (Trivedi & Patel, 2022).

The LSTM architecture is described by the following equations (Song & Choi, 2023):
The forget gate f_t determines which information from the previous cell state C_{t-1} should be forgotten or retained:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f \tag{4}$$

where x_t is the input vector at time t, and σ is the logistic sigmoid function.
The input gate i_t determines the new information to be updated in the cell state C_t:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{5}$$

The candidate cell state $\tilde{C}_t$ to be added is computed as:

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{6}$$

The cell state $\tilde{C}_t$ is updated as:

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{7}$$

where $\odot$ denotes the Hadamard product.
The output gate O_t determines the part of the cell state to be output:

$$O_t = \sigma(W_O \cdot [h_{t-1}, x_t] + b_O) \tag{8}$$

Finally, the output h_t is calculated as:

$$h_t = O_t \odot \tanh(C_t) \tag{9}$$

According to (Hochreiter & Schmidhuber, 1997), these gates control the information to be forgotten or remembered, enabling LSTMs to manage long-term dependencies and predict time series data with varying time steps. LSTMs are also effective in handling noise, distributed representations, and continuous values.

2.3.2. GRU

The Gated Recurrent Unit (GRU) is a streamlined version of the Long Short-Term Memory (LSTM) network, featuring fewer parameters. While LSTM networks include update, input, forget, and output gates along with an internal memory state, GRUs simplify this architecture by using only update and reset gates. This combines the functions of the forget and input gates into a single update gate, reducing the number of tensor operations and resulting in faster training (Song & Choi, 2023).

GRUs merge the cell state and hidden state, making them effective for sequence learning and capable of addressing the vanishing or exploding gradient problems in Recurrent Neural Networks (RNNs) when learning long-term dependencies (Shen *et al.*, 2018). GRUs generally perform better than LSTMs when the training data is limited, while LSTMs excel in retaining longer sequences.

The following equations detail the memory updates for each hidden layer at each time step (Song & Choi, 2023):
The reset gate r_t modulates the influence of the previous hidden state h_{t-1}:

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t]) \tag{10}$$

where x_t represents the current input and h_{t-1} denotes the previous hidden state.
The update gate z_t decides whether the current input x_t should be ignored:

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t]) \tag{11}$$

The candidate activation $\tilde{h}_t$ is computed as:

$$\tilde{h}_t = \tanh(W_h \cdot [r_t \odot h_{t-1}, x_t]) \tag{12}$$

The hidden state h_t is updated using:

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot \tilde{h}_t \tag{13}$$

In these equations, W_r, W_z, and W_h are the weight matrices that are adjusted during training.
Overall, GRUs offer a simpler and more efficient alternative to LSTMs, particularly advantageous when working with smaller datasets.

2.3.3. Hybrid model

Hybrid methods in stock price prediction combine multiple techniques to enhance accuracy. This approach mitigates the limitations of individual methods, reduces the risk of overfitting, and allows for adjustments based on changing market conditions. Consequently, hybrid methods provide more robust and reliable stock price predictions in the face of market complexity and volatility.

The hybrid LSTM-GRU and GRU-LSTM models integrate two types of recurrent neural networks (RNNs): LSTM and GRU. Both networks are designed to address the issues of vanishing gradients and long-term dependencies common in RNNs. By combining LSTM and GRU, these hybrid methods leverage the strengths of each architecture to enhance prediction performance.

LSTM excels at storing and accessing long-term information, making it effective for time series data such as stock price movements. On the other hand, GRU, being simpler than LSTM, trains sequential patterns in the data more efficiently, thereby improving overall network performance (Song & Choi, 2023).

2.4. Bayesian optimization

Bayesian optimization (BO) is a method used to find the global optimum x^* of an objective function f within a search space X (Wang *et a*l., 2023):

$$x^* = \arg\min_{x \in X} f(x) \tag{14}$$

The objective function f can be a black-box function that requires expensive evaluations, such as resource-intensive experiments or complex numerical simulations (Yang *et al.*, 2022). To minimize the number of function evaluations during the search for the optimum, BO combines a simple algorithm with a statistical model that explicitly describes f (Lizotte, 2008). The BO procedure typically involves two main steps. First, it adapts the hidden patterns of the function f based on observed data D using a surrogate model. Second, it optimizes the acquisition function $u(x \mid D)$ based on the posterior distribution and the surrogate model estimate to select the next sample point to be evaluated in the search space X. The acquisition function measures the potential value of evaluating f at specific search points, enabling BO to identify the most promising points for evaluation (Daulton, 2023).

BO is particularly efficient for optimizing hyperparameters by exploring various combinations iteratively. The cost function is computationally expensive due to numerous parameter combinations, and BO aims to find the optimal combination that minimizes the Mean Squared Error (MSE) with the fewest evaluations. This is achieved by selecting combinations that yield the best results in subsequent iterations based on previous evaluations.

Hyperparameter configurations are managed using 'Categorical', 'Real', and 'Integer' objects from the Scikit-Optimize ('skopt') library. The objective function evaluates model performance with TimeSeriesSplit for cross-validation and returns the MSE. The hyperparameters and their respective values are shown in Table 1.

Table 1 Hyperparameters and their values

Activation	tanh, relu, sigmoid, softmax
Optimizer	adam, adamax, rmsprop, sgd
Learning-rate	0.0001, 0.001, 0.05, 0.1
Loss	mean-squared-error
Neuron	32, 64, 128
Dropout-rate	0.1, 0.2

By iterating through these combinations and leveraging BO, the optimal set of hyperparameters can be identified with minimized computational effort and improved model performance.

BO employs two key elements for global optimization

2.4.1. Surrogate model

Surrogate models are utilized to approximate the cost function, thereby reducing computational requirements. Among these models, the Gaussian Process (GP) stands out, providing probabilistic estimates of the cost function and optimized using the ‘gp_minimize’ method. Additionally, Random Forest (RF) and Extra Trees (ET) models employ bagging and random feature selection techniques to estimate the cost function with reduced variance; these models are optimized using the ‘forest_minimize’ method. Furthermore, Gradient Boosting Regression Trees (GBRT) adopt a sequential approach, building trees that iteratively correct errors and model complex relationships, optimized via the ‘gbrt_minimize’ method. Collectively, these surrogate models enable efficient and effective optimization by approximating the cost function, thus facilitating the exploration of optimal solutions with lower computational demands.

1. Gaussian Process (GP)

   One approach to constructing surrogate models is Gaussian Processes (GPs), which provide flexible non-parametric statistical models over a wide range of functional domains. GPs provide information not only about the expected value of a function f but also about its uncertainty. The main idea is to approximate the cost function based on its observed values at some points X_k, assuming that the function values follow a multivariate Gaussian Process with a prior distribution (Diez, n.d.):

$$p(x;\mu,\Sigma)=\frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma|^{\frac{1}{2}}}e^{\left(-\frac{1}{2}(x-\mu)^{\tau}\Sigma^{-1}(x-\mu)\right)} \tag{15}$$

Here, $|\Sigma|$ is the determinant of the covariance matrix with coefficients in the correlation function $K(x_m, x_n, \theta)$. The hyperparameter θ is calibrated based on the maximum likelihood principle, and the kernel selection greatly affects the regression accuracy. The acquisition function, which is faster to evaluate, is used to select the next parameter to be evaluated.

Bayesian optimization with Gaussian Processes involves updating the model based on the reported function evaluations, preserving the posterior model of the target function. Through mathematical transformations and conditional probabilities, the posterior distribution $P(x_{k+1})$ can be estimated, with $f(x_{k+1})$ expressed as a function of $f(x_k)$ and Σ with uncertainty. This posterior model serves as a surrogate Gaussian Process model for the function f.

2. Random Forest (RF)

In Random Forest, each tree in the ensemble is constructed from bootstrap samples taken from the training set. Additionally, when splitting a node during tree construction, the chosen split is not the best split among all features. Instead, the selected split is the best split from a randomly chosen subset of features. This randomness slightly increases the bias of the forest compared to a single non-random tree, but the variance is significantly reduced due to the averaging process. This reduction in variance generally compensates for the increased bias, resulting in an overall more effective model (Diez, n.d.).

3. Extra Trees (ET)

In Extra Trees, the level of randomness in calculating splits is further enhanced compared to random forests. Instead of selecting the best split threshold from all features, a random subset of candidate features is used. For each candidate feature in this subset, thresholds are randomly generated, and the best among these randomly chosen thresholds is selected as the split rule. This approach typically results in a further reduction in model variance, albeit with a slight increase in bias (Diez, n.d.).

4. Gradient Boosted Regression Trees (GBRT)

Gradient Tree Boosting or Gradient Boosted Regression Trees (GBRT) is a generalization of boosting for arbitrary differentiable loss functions. Developed by (Friedman, 2001), GBRT is used in areas such as web search ranking and ecology. GBRT is an accurate and effective method for both regression and classification.

The main goal of this algorithm is to fit a regression tree to the difference between the observed response and the aggregate prediction of all previous learners. A fixed-size decision tree algorithm is chosen as the base learner or weak learner. In boosting, weak learners are trained iteratively to reduce the bias and variance of the predictions. At each iteration, weights are assigned to the weak learners based on their error rates, and each training instance is reweighted according to its error. The final model is a sum of the weighted weak learners, which is used to evaluate the costly function.

GBRT builds an additive model as follows (Diez, n.d.):

$$f(x)=\sum_{k=1}^{K}\gamma_k h_k(x) \tag{16}$$

where $h_k(x)$ is a weak learner. GBRT uses a fixed-size decision tree as a weak learner, which is capable of handling mixed-type data and modeling complex functions. The model is built in a stepwise manner:

$$f_k(x)=f_{k-1}+\gamma_k h_k(x) \tag{17}$$

The new tree h_k tries to minimize the loss L given the previous ensemblef_{k-1}:

$$h_k=\arg\min_{\mathrm{h}}\Sigma_{i=1}^{n}L\big(y_i, f_{k-1}(x_i)+h(x_i)\big) \tag{18}$$

The initial model f_0 is usually the average of the target values for least-squares regression. GBRT accomplishes this minimization via steepest descent, where the direction of the steepest descent is the negative gradient of the loss function evaluated on the current model F_{m-1}:

$$f_k(x)=f_{k-1}(x)-\gamma_k\Sigma_{i=1}^{n}\nabla_f L\big(y_i, f_{k-1}(x_i)\big) \tag{19}$$

The step length γ_k is chosen using a line search:

$$\gamma_k=\arg\min_{\gamma}\Sigma_{i=1}^{n}L\left(y_i, f_{k-1}(x_i)-\gamma\frac{\partial L\big(y_i, f_{k-1}(x_i)\big)}{\partial f_{k-1}(x_i)}\right) \tag{20}$$

Algorithms for regression and classification differ only in the loss function used. In sequential optimization using GBRT, evaluation is performed in two ways: if x_0 is provided but not $f(x_0)$, the elements of x_0 are evaluated first, followed by a number of random starts evaluations (n random starts). Finally, (n calls - len(x_0) - n random starts) evaluations are performed with the guidance of the surrogate model. If both x_0 and $f(x_0)$ are provided, random starts evaluations are performed first, then (n calls - n random starts) evaluations are performed with the guidance of the surrogate model.

2.4.2. Acquisition function

The acquisition function is a utility function that guides the search to reach the optimum of the objective function by identifying the next sample location, which is crucial in Bayesian optimization (Wang *et al*., 2023). In this study, the Expected Improvement (EI) acquisition function is used. EI measures the expected value of the improvement over the best observed solution $f(x^*)$ by evaluating the objective function at a specific point x.

Using EI with surrogate models such as Gaussian Process (GP), Random Forest (RF), Extra Trees (ET), and Gradient Boosted Regression Trees (GBRT) follows the same principle: maximizing the expected improvement. GP constructs a probabilistic model of the objective function f, providing estimates of the mean $\mu(x)$ and uncertainty $\sigma(x)$ at point x. EI for GP is calculated as:

$$EI(x) = \mu(x) - f(x^*)\Phi\left(\frac{\mu(x) - f(x^*)}{\sigma(x)}\right) + \sigma(x)\phi\left(\frac{\mu(x) - f(x^*)}{\sigma(x)}\right) \tag{21}$$

RF, ET, and GBRT regression models yield different estimates for $\mu(x)$ and $\sigma(x)$. While the calculation details may vary, the basic EI concept remains the same across these models.

Each optimization process uses `n_call=50` to determine the number of evaluations required to find the best hyperparameter combination. The `skopt` library functions—`gp_minimize`, `forest_minimize`, and `gbrt_minimize`—are tailored to different models, ensuring effective hyperparameter optimization.

2.5. Performance metrics

In this study, evaluation metrics such as MSE, RMSE, MAE, MAPE, and R-squared are used to compare the performance of the proposed models for predicting stock price data. These metrics are calculated as follows (Chen *et al.*, 2023)(Kim & Jang, 2023):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - y_i')^2 \tag{22}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - y_i')^2} \tag{23}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - y_i'|^2 \tag{24}$$

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - y_i'}{y_i}\right| \times 100\% \tag{25}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - y_i')^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{26}$$

where y_i is the actual value, y_i' is the predicted value n is the number of data points, and $\bar{y}$ is the mean of the actual y_i values.

3. Results and discussions

3.1. Model performances

Gaussian Process (GP)

Table 2 The results of MSE with their best parameters for BO-GP

Models	Neurons	Activation	Dropout	Optimizer	Lr	MSE
LSTM	128	tanh	0.1	adam	0.001	0.000223
GRU	128	tanh	0.1	adam	0.001	0.000255
LSTM-GRU	128	tanh	0.2	adam	0.0001	0.000299
GRU-LSTM	128	tanh	0.1	adam	0.001	**0.000183**

Table 2 shows the MSE results of the LSTM, GRU, and hybrid models (LSTM-GRU and GRU-LSTM) using optimized parameters. All four models have the same number of neurons (128), tanh activation function, and use the adam optimizer with varying learning rates. The GRU-LSTM model, with parameters neuron = 128, activation = tanh, dropout = 0.1, optimizer = adam, and learning rate = 0.001 (epoch = 100), has the lowest MSE of 0.000183. This is followed by the LSTM model with an MSE of 0.000223, the GRU model with an MSE of 0.000255, and the LSTM-GRU model with an MSE of 0.000299.

Table 3 BO-GP with LSTM, GRU, and hybrids

Error Metrics	LSTM	GRU	LSTM-GRU	GRU-LSTM
MSE	0.000223	0.000255	0.000299	**0.000183**
RMSE	0.014921	0.015965	0.017289	**0.013518**
MAE	0.011623	0.012861	0.013333	**0.010278**
MAPE	1.44%	1.59%	1.66%	**1.28%**
R-squared	0.967267	0.962526	0.956053	**0.973133**

Table 3 shows the performance evaluation results of the LSTM, GRU, and hybrid models (LSTM-GRU and GRU-LSTM) on the test data using error metrics including MSE, RMSE, MAE, MAPE, and R2. The GRU-LSTM model demonstrates the best performance in reducing errors, with lower MSE, RMSE, MAE, and MAPE values compared to the other models. Additionally, the GRU-LSTM model has a higher R-squared value of 0.973133, indicating superior prediction capabilities.

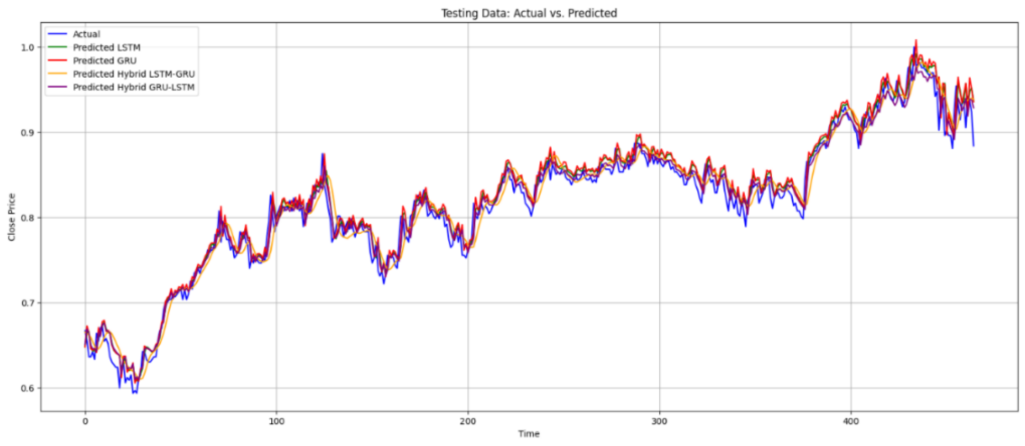

Figure 1 Plot of actual vs. predicted values for LSTM, GRU, and hybrid models using Gaussian Process (GP)

Random Forest (RF)

Table 4 The results of MSE with their best parameters for BO-RF

Models	Neurons	Activation	Dropout	Optimizer	Lr	MSE
LSTM	32	relu	0.1	adam	0.0001	0.000539
GRU	64	relu	0.2	adam	0.001	0.000277
LSTM-GRU	64	tanh	0.2	adam	0.001	**0.00023**
GRU-LSTM	128	relu	0.2	adamax	0.0001	0.000391

Based on Table 4, the LSTM-GRU model with parameters neuron = 64, activation = tanh, dropout = 0.2, optimizer = adam, and learning rate = 0.001 (epoch = 100) produces the lowest MSE of 0.00023.

Table 5 BO-RF with LSTM, GRU, and hybrids

Error Metrics	LSTM	GRU	LSTM-GRU	GRU-LSTM
MSE	0.000539	0.000277	**0.00023**	0.000391
RMSE	0.023214	0.016648	**0.015163**	0.019764
MAE	0.018273	0.01284	**0.011685**	0.01528
MAPE	2.26%	1.60%	**1.44%**	1.90%
R-squared	0.920769	0.959253	**0.966198**	0.942569

Table 5 shows the evaluation results for the LSTM, GRU, and hybrid models (LSTM-GRU and GRU-LSTM). The LSTM-GRU model demonstrates the best performance, with the lowest MSE of 0.00023, RMSE of 0.015163, MAE of 0.011685, MAPE of 1.44%, and the highest R^2 value of 0.966198.

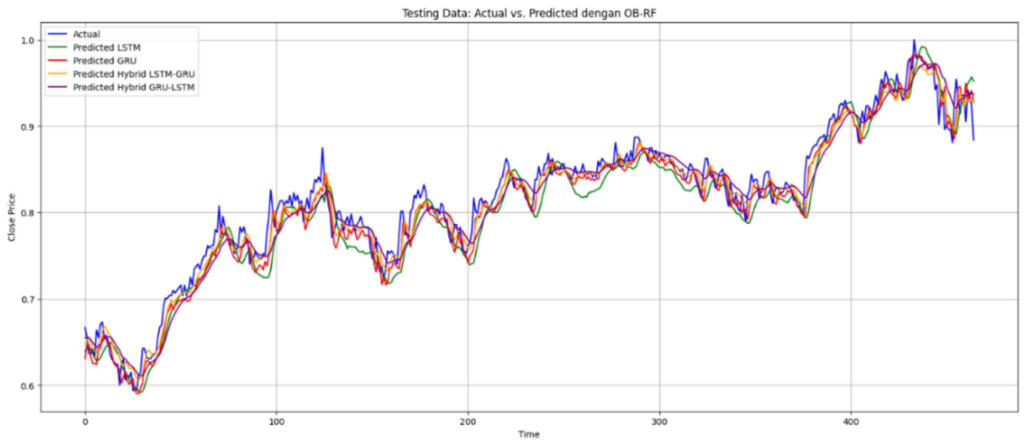

Figure 2 Plot of actual vs. predicted values for LSTM, GRU, and hybrid models using Random Forest (RF)

Extra Trees (ET)

Table 6 The results of MSE with their best parameters for BO-ET

Models	Neurons	Activation	Dropout	Optimizer	Lr	MSE
LSTM	32	tanh	0.1	adam	0.0001	0.000493
GRU	128	tanh	0.1	adam	0.001	**0.000188**
LSTM-GRU	128	relu	0.2	adam	0.001	0.000308
GRU-LSTM	64	relu	0.1	adam	0.001	0.000212

Table 6. shows the MSE results for the four models with their best parameters. The GRU model, with parameters neuron = 128, activation = tanh, dropout = 0.1, optimizer = adam, and learning rate = 0.001, demonstrates the best performance, achieving the lowest MSE of 0.000188.

Table 7 BO-ET with LSTM, GRU, and hybrids

Error Metrics	LSTM	GRU	LSTM-GRU	GRU-LSTM
MSE	0.000493	**0.000188**	0.000308	0.000212
RMSE	0.022212	**0.013721**	0.017539	0.014551
MAE	0.017697	**0.010148**	0.013749	0.011105
MAPE	2.17%	**1.25%**	1.70%	1.38%
R-squared	0.927463	**0.972322**	0.954771	0.968871

The model evaluation results in Table 7 indicate that the GRU model with optimal parameters— neuron = 128, activation = tanh, dropout = 0.1, optimizer = adam, and learning rate = 0.001 (epoch = 100)—exhibits the best performance among all models tested. The GRU model achieves the lowest MSE of 0.000188, with an RMSE of 0.013721, MAE of 0.010148, and MAPE of 1.25%. The R2 value for this model is 0.972322, reflecting a very good ability to explain data variability, with an explanation accuracy of 97.23%.

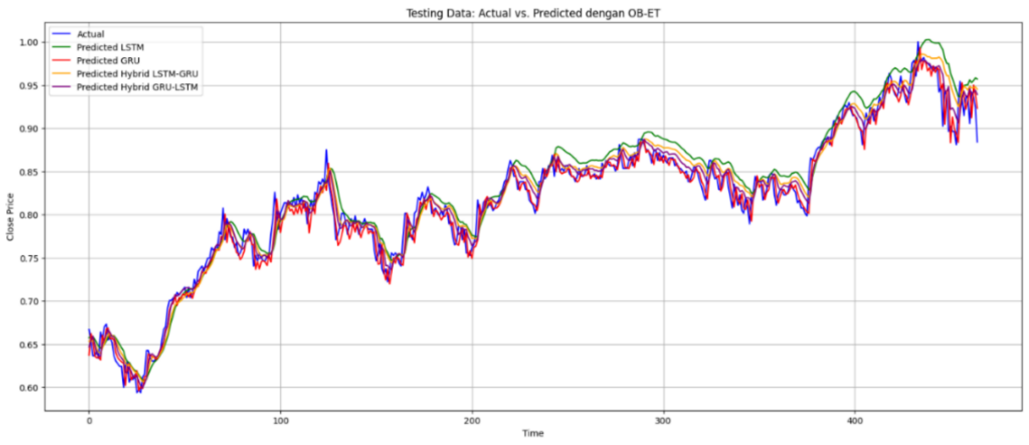

Figure 3 Plot of actual vs. predicted values for LSTM, GRU, and hybrid models using Extra Trees (ET)

Gradient Boosted Regression Trees (GBRT)

Table 8 The results of MSE with their best parameters for BO-GBRT

Models	Neurons	Activation	Dropout	Optimizer	Lr	MSE
LSTM	32	tanh	0.1	adam	0.001	0.000218
GRU	64	tanh	0.2	adam	0.001	0.000204
LSTM-GRU	64	tanh	0.1	adam	0.001	**0.000187**
GRU-LSTM	64	tanh	0.1	adam	0.001	0.000338

Based on Table 8, the LSTM-GRU model with parameters neuron = 64, activation = tanh, dropout = 0.1, optimizer = adam, and learning rate = 0.001 yields the best results, achieving an MSE of 0.000187.

Table 9 BO-GBRT with LSTM, GRU, and hybrids

Error Metrics	LSTM	GRU	LSTM-GRU	GRU-LSTM
MSE	0.000218	0.000204	**0.000187**	0.000338
RMSE	0.014766	0.014296	**0.013685**	0.018384
MAE	0.011211	0.010716	**0.01042**	0.015034
MAPE	1.39%	1.32%	**1.29%**	1.82%
R-squared	0.967944	0.969952	**0.972467**	0.950308

Table 9 presents the performance evaluation results on the test data, highlighting significant differences between the LSTM, GRU, and hybrid models (LSTM-GRU and GRU-LSTM). The LSTM-GRU model demonstrates the best performance with an MSE of 0.000187, RMSE of 0.013685, MAE of 0.01042, MAPE of 1.29%, and R^2 of 0.972467.

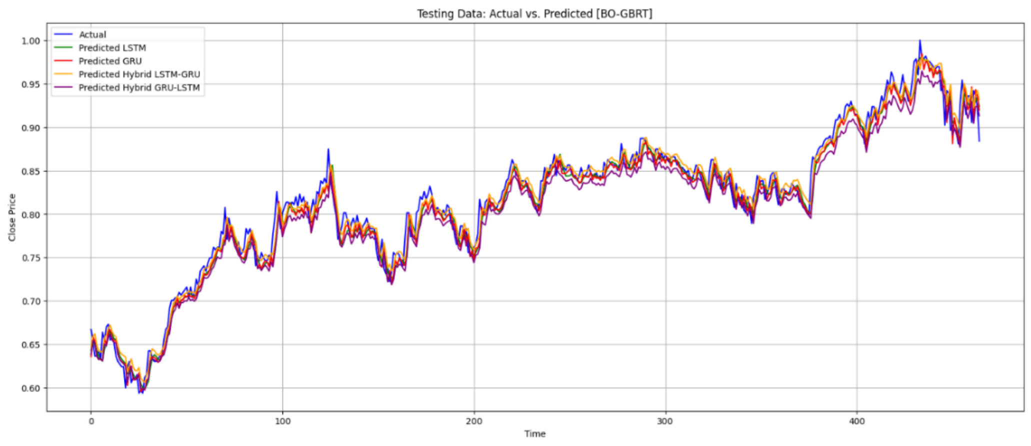

Figure 4 Plot of actual vs. predicted values for LSTM, GRU, and hybrid models using Gradient Boosted Regression Trees (GBRT)

3.2. Comparison models

Table 10 BO results with different surrogate functions

Model Pengganti	Error Metrics	LSTM	GRU	LSTM-GRU	GRU-LSTM
	MSE	0.000223	0.000255	0.000299	**0.000183**
	RMSE	0.014921	0.015965	0.017289	**0.013518**
GP	MAE	0.011623	0.012861	0.013333	**0.010278**
	MAPE	1.44%	1.59%	1.66%	**1.28%**
	R-squared	0.967267	0.962526	0.956053	**0.973133**
	MSE	0.000539	0.000277	**0.00023**	0.000391
	RMSE	0.023214	0.016648	**0.015163**	0.019764
RF	MAE	0.018273	0.01284	**0.011685**	0.01528
	MAPE	2.26%	1.60%	**1.44%**	1.90%
	R-squared	0.920769	0.959253	**0.966198**	0.942569
	MSE	0.000493	**0.000188**	0.000308	0.000212
	RMSE	0.022212	**0.013721**	0.017539	0.014551
ET	MAE	0.017697	**0.010148**	0.013749	0.011105
	MAPE	2.17%	**1.25%**	1.70%	1.38%
	R-squared	0.927463	**0.972322**	0.954771	0.968871
	MSE	0.000218	0.000204	**0.000187**	0.000338
	RMSE	0.014766	0.014296	**0.013685**	0.018384
GBRT	MAE	0.011211	0.010716	**0.01042**	0.015034
	MAPE	1.39%	1.32%	**1.29%**	1.82%
	R-squared	0.967944	0.969952	**0.972467**	0.950308

The results of the model performance evaluation with various surrogate models for Bayesian optimization reveal significant variations in error metrics for the LSTM, GRU, and hybrid models (LSTM-GRU and GRU-LSTM). Based on Table 10, the model with a Gaussian Process (GP) surrogate shows that the GRU-LSTM model achieves the best performance with an MSE of 0.000183, RMSE of 0.013518, MAE of 0.010278, MAPE of 1.28%, and R^2 of 0.973133. The Random Forest (RF) surrogate model shows that the LSTM-GRU model performs best with an MSE of 0.000230, RMSE of 0.015163, MAE of 0.011685, MAPE of 1.44%, and R^2 of 0.966198. The Extra Trees (ET) surrogate yields the best results for the GRU model with an MSE of 0.000188, RMSE of 0.013721, MAE of 0.010148, MAPE of 1.25%, and R^2 of 0.972322. Meanwhile, the Gradient Boosted Regression Trees (GBRT) surrogate shows that the LSTM-GRU model achieves the best results with an MSE of 0.000187, RMSE of 0.013685, MAE of 0.01042, MAPE of 1.29%, and R^2 of 0.972467.

These findings highlight that the choice of surrogate model significantly impacts the performance of the primary model in capturing time series data patterns. The GRU-LSTM model with a GP surrogate demonstrates excellent performance, indicating its effectiveness in capturing data complexity.

This analysis underscores the benefits of using hybrid models, particularly in leveraging the strengths of each memory unit (LSTM and GRU), and emphasizes that selecting the appropriate surrogate model can substantially enhance prediction accuracy.

4. Conclusion

Based on the research results and discussion above, it can be concluded that the selection of appropriate parameters and surrogate models in Bayesian optimization significantly affects the performance of the main model, particularly for time series data. The hybrid LSTM-GRU and GRU-LSTM models exhibit superior performance compared to single models, with GRU-LSTM optimized using Gaussian Process (GP) demonstrating the best overall performance. The LSTM-GRU model also yields very good results when optimized using Random Forest (RF) and Gradient Boosted Regression Trees (GBRT), demonstrating that combining architectures can substantially enhance prediction accuracy. These findings emphasize the importance of selecting an appropriate surrogate model to improve prediction accuracy and indicate that this approach can be effectively applied to various practical applications utilizing time series data, such as gold price prediction, demand forecasting, and economic trend analysis.

References

Baek, H. (2023). A CNN-LSTM Stock Prediction Model Based on Genetic Algorithm Optimization. *Asia-Pacific Financial Markets*, *0123456789*. https://doi.org/10.1007/s10690-023-09412-z

Chen, C., Xue, L., & Xing, W. (2023). Research on Improved GRU-Based Stock Price Prediction Method. *Applied Sciences (Switzerland)*, *13*(15). https://doi.org/10.3390/app13158813

Daulton, S. J. (2023). *Bayesian Optimization in Adverse Scenarios*.

Diez, M. (n.d.). *Porosity Optimization in Nanoporous materials via Machine Learning*. Politecnico di Torino.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, *29*(5), 1189–1232. https://doi.org/10.1214/aos/1013203451

Gur, Y. E. (2024). Comparative Analysis of Deep Learning Models for Silver Price Prediction: CNN, LSTM, GRU and Hybrid Approach. *Akdeniz Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, *24*(1), 1–13. https://doi.org/10.25294/auiibfd.1404173

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Kervanci, I. S., Akay, M. F., & Özceylan, E. (2024). Bitcoin Price Prediction Using Lstm, Gru and Hybrid Lstm-Gru With Bayesian Optimization, Random Search, and Grid Search for the Next Days. *Journal of Industrial and Management Optimization*, *20*(2), 570–588. https://doi.org/10.3934/jimo.2023091

Kim, G. Il, & Jang, B. (2023). Petroleum Price Prediction with CNN-LSTM and CNN-GRU Using Skip-Connection. *Mathematics*, *11*(3). https://doi.org/10.3390/math11030547

Lizotte, D. J. (2008). *Practical Bayesian Optimization*.

Shen, G., Tan, Q., Zhang, H., Zeng, P., & Xu, J. (2018). Deep learning with gated recurrent unit networks for financial sequence predictions. *Procedia Computer Science*, *131*, 895–903. https://doi.org/10.1016/j.procs.2018.04.298

Song, H., & Choi, H. (2023). Forecasting Stock Market Indices Using the Recurrent Neural Network Based Hybrid Models: CNN-LSTM, GRU-CNN, and Ensemble Models. *Applied Sciences (Switzerland)*, *13*(7). https://doi.org/10.3390/app13074644

Trivedi, D. V., & Patel, P. S. (2022). An Analysis of GRU-LSTM Hybrid Deep Learning Models for Stock Price Prediction. *International Journal of Scientific Research in Science, Engineering and Technology*, *4099*, 47–51. https://doi.org/10.32628/ijsrset229264

Wang, X., Jin, Y., Schmitt, S., & Olhofer, M. (2023). Recent Advances in Bayesian Optimization. *ACM Computing Surveys*, *55*(13s). https://doi.org/10.1145/3582078

Yang, S., Liu, B., Hong, Z., & Zhang, Z. (2022). Bayesian Optimization-Based Beam Alignment for MmWave MIMO Communication Systems. *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC*, *2022-Septe*(i), 825–830. https://doi.org/10.1109/PIMRC54779.2022.9977858

Zhang, J., Ye, L., & Lai, Y. (2023). Stock Price Prediction Using CNN-BiLSTM-Attention Model. *Mathematics*, 1–18. https://doi.org/https:// doi.org/10.3390/math11091985

Zhao, C., Wu, M., Liu, J., Duan, Z., li, J., Shen, L., Shangguan, X., Liu, D., & Wang, Y. (2023). Progress and prospects of data-driven stock price forecasting research. *International Journal of Cognitive Computing in Engineering*, *4*, 100–108. https://doi.org/10.1016/j.ijcce.2023.03.001