Vol. 11 No.2 Tahun 2025, pp 201-211
Jurnal Teknologi dan Manajemen Informatika
http:// http://jurnal.unmer.ac.id/index.php/jtmi
P-ISSN: 1693-6604 E-ISSN: 2580-8044

Implementation of the CNN-LSTM Hybrid Model in Predicting Bitcoin Price Fluctuations

Candra Wibowo[1], Ronsen Purba[2], Muhammad Fermi Pasha[3]

Informatics Faculty, Mikroskil University, Indonesia

Article Info

Article History
Received : 30-09-2025
Revised : 17-11-2025
Accepted : 05-12-2025

Keywords
Price Prediction;
Bitcoin;
CNN-LSTM;
Deep Learning;
Cryptocurrency;
Time Series;
RMSE;
MAPE;
MAE

Corresponding Author
Candra Wibowo,
Informatics Faculty,
Mikroskil University
candrawibowo2018@gmail.com

ABSTRACT

Digital financial systems of today face formidable obstacles from the extreme price volatility and unpredictability of Bitcoin. Data cleaning, min-max normalization, and sequence creation with a sliding window were performed on the daily BTC-USD historical data received from Yahoo Finance from 2020 to 2024 before implementing a hybrid Convolutional Neural Network-Long Short-Term Memory (CNN-LSTM) model in this study. The CNN layers are responsible for extracting local patterns with a limited time horizon, whereas the LSTM layers are responsible for capturing the time series' long-term relationships. The experimental findings show that the CNN-LSTM model outperforms the CNN and LSTM in terms of predictive ability, with an RMSE of 2,202.717, an MAE of 1,553.202, and a MAPE of 2.244%, which translates to an accuracy of about 97.756%. These results provide useful information for adaptive trading techniques and digital asset risk management based on artificial intelligence, and they prove that the hybrid method is successful in dealing with complicated, non-linear, and unpredictable trends in the cryptocurrency market.

INTRODUCTION

The rise of cryptocurrencies is just one example of how the global financial system has been profoundly altered by developments in digitization and financial technology. Bitcoin has the largest market capitalization and has been the most influential cryptocurrency so far [1]. Bitcoin is decentralized, not regulated by any authority, and is widely used both as a medium of exchange and as a digital investment [2]. Nevertheless, Bitcoin prices are highly volatile. Their movements are influenced by multiple factors such as market demand, regulations, global news, and the activities of large investors (whales) [3], [4]. This high volatility makes Bitcoin price prediction particularly challenging, especially for investors seeking to maximize returns or minimize risks [4].

Several methods have been devised in an attempt to predict the price of Bitcoin. The complexities and non-linearities of bitcoin price data have proven to be too much for traditional approaches like ARIMA and linear regression to handle [5]. For this reason, methods based on deep learning and machine learning, such as LSTM and RNN models, have become increasingly popular [6], [7]. Although LSTM excels at capturing time series data long-term dependencies, it might struggle to extract notable initial characteristics of the hyperparameter

This is an open access article under the CC BY-SA license.
DOI: https://doi.org/10.26905/jtmi.v11i2.16239

setup isn't fine-tuned [8]. On the other hand, Convolutional Neural Networks (CNN) excel at extracting important features from visual patterns or grid-structured data such as price charts [9]. Accordingly, the integration of these two models in the form of a hybrid CNN-LSTM is believed to improve accuracy in cryptocurrency price prediction [10], [11].

Since it can capture both spatial and temporal data concurrently, the hybrid CNN-LSTM model beats separate models, according to several studies [12]. Essential components of a strong prediction model include normalization and other preprocessing methods, as well as assessment measures such as Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE) [13], [14]. In order to estimate the price of Bitcoin, Jha and Yadav (2022) created a hybrid deep learning model that combines CNN and LSTM. They found that combining the two designs improved predicted accuracy over either CNN or LSTM alone. The model takes use of CNN's power to identify regional trends and LSTM's capacity to record the sequential order of Bitcoin price data from the past [15]. On top of that, using time series data, Sharma et al. (2022) projected Bitcoin values using a hybrid CNN-LSTM Model. Their research proved that this combined strategy greatly cut down on prediction mistakes and performed better even when market conditions were unpredictable [16].

In their work Comparative Analysis of Deep Learning Models, Kim et al. (2021) also used a comparison technique to assess the performance of different architectures, such as CNN, LSTM, and hybrid models. When it came to accuracy, resistance to data noise, and training efficiency, their results demonstrated that the CNN-LSTM model consistently performed better [17]. Additionally, Jang and Lee (2020) introduced a Bitcoin price prediction method based on bayesian neural networks, offering a probabilistic approach to forecasting. While this model demonstrated strengths in uncertainty estimation, its predictive accuracy remained less competitive compared to the CNN-LSTM model [18]. Although more recent studies such as Tiwari and Kumar (2023) and Omole and Enke (2024) have been conducted, they have not specifically examined the full period of 2020–2024, which encompasses one halving cycle [10], [19]. In order to fully validate the performance of the hybrid CNN-LSTM model, this study uses the most recent dataset from 2020–2024. The purpose of this work is to fill this knowledge gap by applying the hybrid CNN-LSTM model. Two validation techniques were applied in this study: loss curve analysis and Time Series Cross-Validation (TSCV).

METHODS

Gathering Bitcoin price history, preprocessing, modeling with a CNN-LSTM architecture, and testing and evaluating the model were the four crucial steps in this work. An 8 GB RAM, 256 GB SSD, and 2.3 GHz Quad-Core Intel Core i5 laptop running macOS Ventura 13.7.1 was used for the investigation. Python and its libraries, Visual Studio Code, Google Colaboratory, and the flowchart designing tool Whimsical.io were all part of the software ecosystem. Two validation techniques were applied: loss curve analysis and TSCV.

The loss curves were examined to observe convergence and detect possible overfitting or underfitting, where a steadily decreasing and flattening curve indicated stable learning. Meanwhile, TSCV repeatedly split the time-series data into sequential training and testing sets to evaluate how well the model generalizes to unseen, dynamic data. The data preprocessing stage aims to prepare the dataset before it is used for model training. The preprocessing steps include data normalization using min-max scaling, where all feature values are transformed into a consistent numerical range to ensure balanced input for the model [20]. The formula used is presented below.

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Description:

X	: the original (raw) value from the dataset
X_min	: the smallest value of the entire data in that feature
X_max	: the largest value of the entire data in that feature
X_norm	: the normalized value within the range [0, 1]

There were two sets extracted from the preprocessed dataset: one for training and one for testing. Predefined hyperparameters (epoch, batch size, and learning rate) were used to train the CNN-LSTM model. During training, the CNN layers retrieved spatial features from pricing patterns, while the LSTM layers caught temporal correlations in the time series data. To analyze the findings, we first measured the model's performance and then plotted the data. The sequence creation process in this study was carried out using a sliding window technique to prepare time-series data for the LSTM model. A window size (timestep) of 10 was used. This study's dataset, which includes historical Bitcoin to US Dollar (BTC-USD) pricing data from January 1, 2020, to December 31, 2024, was sourced from the Yahoo Finance website.

There are 1,825 data points in the dataset that were utilized for this investigation. The yfinance library, an open-source tool for retrieving market and financial asset data from Yahoo Finance, was used to collect data in this study via the Python programming language. The dataset was retrieved by calling the download function with parameters including the asset symbol ("BTC-USD"), time range, and data interval. The resulting data, comprising open, high, low, close, and trading volume, was stored in CSV format and used as the primary dataset for model training and testing. The entire process was automated through Python scripts to ensure efficiency and consistency. Figure 1 shows the CNN-LSTM model flowchart.

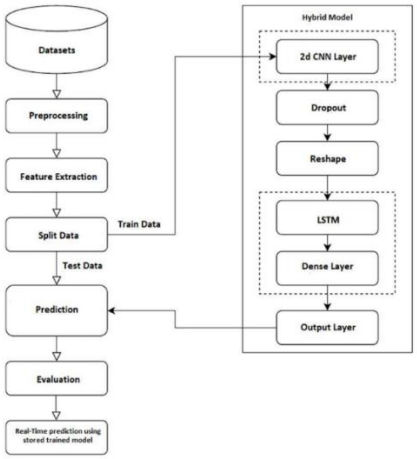


Figure 1. CNN-LSTM Model Flowchart

Based on the model architecture, the single CNN model consists of two main layers: one Conv1D layer with 64 filters and a kernel size of 2, followed by one MaxPooling1D layer. The baseline LSTM model uses a single LSTM layer with 50 units. In contrast, the hybrid CNN-LSTM model combines two Conv2D layers with 32 and 64 filters using 3x3 kernels, followed by one LSTM layer with 128 units. The hybrid structure works by allowing the CNN block to first extract local patterns from each subsequence, then reshaping the CNN output and passing it to the LSTM layer to learn long-term temporal dependencies. This combination enables the model to capture both short-term local features and long-range temporal patterns, resulting in more accurate Bitcoin price predictions.

After the training process is completed, the model is tested using the testing dataset. The performance evaluation is carried out using regression metrics, including Root Mean Squared Error (RMSE). The formula used is presented below.

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

In addition, testing was also carried out using MAE. The formula used is presented below.

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Testing using the Mean Absolute Percentage Error (MAPE) was also conducted, in which:

$$\text{MAPE} = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|$$

n = number of data points
y_i = original (actual) data value at time *i*
y_i = the predicted value of the data at time *i*.

The MAPE value can be interpreted into four categories:

1. < 10% = very accurate
2. 10 – 20% = good
3. 20 – 50% = fair
4. > 50% = not accurate

RESULTS AND DISCUSSION

Since Bitcoin prices are so volatile, we used the Min-Max Scaling approach to normalize the data, which puts all values in the range of 0 to 1. This step, when executed with scikit-learn's MinMaxScaler, increases training efficiency by decreasing the likelihood of gradient vanishing, improving optimization stability, and speeding up convergence. Table 1 shows the data following normalization, is summarized statistically.

Table 1. Dataset after Normalization

	Open	**High**	**Low**	**Close**	**Adj Close**	**Volume**
count	1827.000000	1827.000000	1827.000000	1827.000000	1827.000000	1827.000000
mean	0.309162	0.308340	0.309713	0.309841	0.309841	0.080569
std	0. 208521	0. 209463	0. 203688	0. 208756	0. 208756	0. 056463
min	0. 000000	0. 000000	0. 000000	0. 000000	0. 000000	0. 000000
25%	0. 144266	0. 143649	0. 149620	0. 144800	0. 144800	0. 044194
50%	0. 268292	0. 271966	0. 266080	0. 269008	0. 269008	0. 071194
75%	0. 451543	0. 452413	0. 444391	0. 452163	0. 452163	0. 100830
max	1.0000000	1.0000000	1.0000000	1.0000000	1.0000000	1.0000000

Based on the table, it can be observed that the minimum values of all attributes are 0, while the maximum values reach 1. This indicates that the normalization process successfully transformed the original value range into the 0–1 interval. The information on mean, standard deviation (std), and quartiles (25%, 50% and 75%) provides an overview of the distribution of the normalized data and ensures that no values fall outside the specified range. Because LSTM requires sequential input, historical Bitcoin prices were transformed into windowed sequences using a sliding window technique. With this method, n past data points predict the next value: for example, a window size of 10 uses 10 days of prices to forecast the 11th day. Table 2 presents the summary of the resulting sequences.

Table 2. Sequence Summary

Description	**Value**
Number of Input Rows (X)	1.817
Number of Features per Sequence	10
Number of Target Outputs (y)	1.817

The input dataset (X) has the shape (1817, 10), indicating 1,817 sequences, each containing 10 historical price data points. The target dataset (y) has the shape (1817), meaning each sequence corresponds to one prediction target, namely the price at the 11th period. This sequence format aligns with the LSTM model's requirement for sequential input data. The original dataset contained ten input features, but not all of them contributed meaningfully to the

model's performance. A feature reduction process was then applied to identify and retain only the most informative variables. By selecting six key features, the model became simpler, faster to train, and less prone to overfitting, while still maintaining strong predictive performance. To maintain the necessary chronological order for time series modeling, the dataset was partitioned according to time. With 1,828 observation rows in total, the Bitcoin price dataset utilized in this study is based on the preprocessing findings. With an 80%-20% split applied, 1,462 rows were designated for training and 366 rows for testing. Table 3 shows the dataset split is summarized.

Table 3. Dataset Split

Dataset Type	Number of Observations	Percentage
Total Dataset (Before Split)	1.828 rows	100%
Training Set	1.462 rows	80%
Testing Set	366 rows	20%

It was now possible to begin modeling the Bitcoin dataset after its preprocessing and splitting. Starting with an input layer that takes in sequences with a certain window size, the CNN-LSTM architecture was created in stages. After reducing dimensionality and extracting local features using CNN and MaxPooling1D layers, stacked LSTM layers capture temporal relationships. Predictions are then generated when the output is flattened and regularized in the Dense and Dropout layers. Also created for the sake of comparison were two baseline models: one was an LSTM meant to capture sequential dependencies, and the other was a standalone CNN that used Conv1D, MaxPooling1D, and Dense layers to identify local patterns. Tabulated in Table 4 is the comprehensive design of the CNN baseline model.

Table 4. Architecture of the CNN Baseline Model

Layer	Output Shape	Parameter
Conv1D	(None, 29, 64)	192
MaxPooling1D	(None, 14, 64)	0
flatten1	(None, 896)	0
dense_3	(None, 50)	44850
dense_4	(None, 1)	51

Contrarily, the LSTM model was specifically designed to assess the capabilities of long-term memory in time-series detection without a convolutional filter. Not like CNN, LSTM is designed to learn linearly, making it more effective in capturing multi-layer temporal dependencies. There are still three dimensions to the data used by LSTM: sample, time, and feature. The basic architecture of LSTM uses prediction output by performing one layer of LSTM.

Table 5. Architecture of the LSTM Baseline Model

Layer	Output Shape	Parameter
lstm_1	(None, 50)	10400
dense_5	(None, 1)	51

The advantage of the hybrid CNN-LSTM approach lies in its ability to handle very long input sequences by dividing them into smaller subsequences. CNN extracts local patterns within each subsequence, while LSTM integrates these interpretations to capture long-term dependencies. In this architecture, each input sample is divided into subsequences of predefined timesteps, processed by CNN blocks wrapped in a TimeDistributed layer to ensure parallel convolution across subsequences. The extracted features are then combined and passed to LSTM to produce the final prediction. The model was implemented in Google Colaboratory using Keras with a TensorFlow backend, enabling modular deep learning architecture design. Key parameters—such as the number of CNN filters, kernel size, pooling size, LSTM units, activation functions, optimizer, learning rate, batch size, and epochs—were determined through

initial experiments and literature review. Table 6 shows the detailed CNN-LSTM architecture is presented.

Table 6. Architecture of the CNN-LSTM Model

Layer	Output Shape	Number of Parameters	Description
Input	(None, 30, 6, 1)	0	Window size 30 with 6 input features
Conv2D	(None, 30, 6, 32)	320	Kernel size 3×3, ReLU activation, generating 32 filters
Conv2D_1	(None, 30, 6, 64)	18,496	Kernel size 3×3, ReLU activation, generating 64 filters
Dropout	(None, 30, 6, 64)	0	Dropout rate of 0.2 to reduce overfitting
Reshape	(None, 30, 384)	0	Reshapes the data to match LSTM input format
LSTM	(None, 128)	262,656	128 units LSTM, captures long-term temporal dependencies
Dense	(None, 64)	8,256	Fully connected layer with 64 neurons, ReLU activation
Dense_1	(None, 1)	65	Output layer, 1 neuron with linear activation

The CNN-LSTM model architecture used in this study consists of several sequential layers designed to extract features from historical Bitcoin price data and capture temporal dependencies. The input layer has a shape of (30, 6, 1), representing a 30-day window with six normalized features. Convolutional layers (Conv2D and Conv1D) with kernel sizes of 3–5 and ReLU activation are applied to detect short-term local patterns, followed by MaxPooling1D with a pool size of 2 to reduce feature dimensions and avoid excessive complexity. The output is then reshaped and passed to multiple LSTM layers with 128 units, some configured with return_sequences=True to preserve timestep outputs.

These LSTM layers capture long-term dependencies in the data. Dropout layers with rates between 0.2 and 0.5 are applied after convolutional and dense layers to prevent overfitting. Dense layers with 128, 64, and 32 neurons, all using ReLU activation, further transform features before the final dense output layer with one neuron and linear activation generates the predicted Bitcoin price. Overall, the architecture trains 289,793 parameters across convolutional, LSTM, and fully connected layers. During training, both training loss and validation loss were monitored to evaluate model stability. As shown in Figure 2, the loss decreased significantly in the initial epochs before stabilizing near a minimum value. The small gap between training and validation loss indicates no significant overfitting, a trend also confirmed in Figure 3 over 50 epochs.

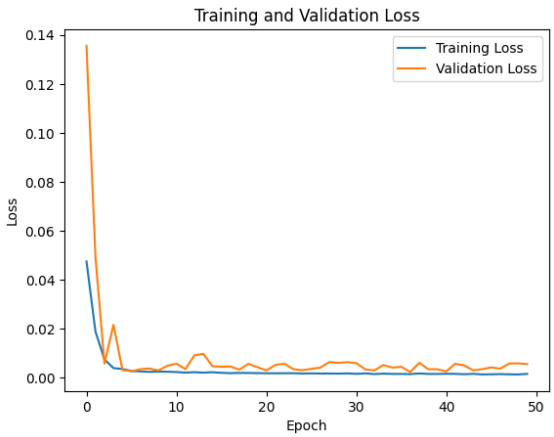


Figure 2. Training and Validation Loss of CNN-LSTM Model

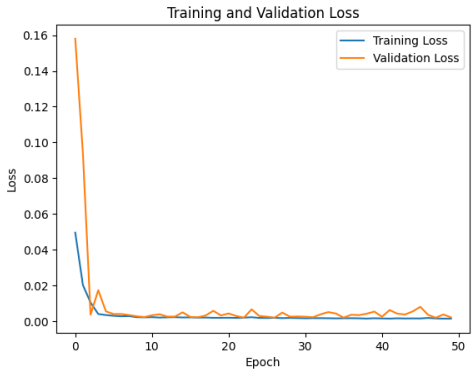


Figure 3. Training and Validation Loss of CNN-LSTM Model on Second try

The difference between the first and second graphs lies in the stability of the validation loss. In the first graph, the validation loss shows slight fluctuations near zero but remains stable without significant signs of overfitting. A similar pattern is observed in the second graph, where the fluctuations of validation loss remain relatively small and align closely with the training loss. This indicates that the model has good generalization capability on the test data and does not experience excessive overfitting.

Overall, the results of both tests demonstrate that the CNN-LSTM model performs training effectively, as reflected by the low loss values and the consistency between training and validation loss patterns. Thus, the model is considered stable in learning the time series data patterns used. The predicted closing values are then compared with the actual values to assess the model's accuracy. This comparison is presented in graphical form to make it easier to observe differences between the actual price trends and the predicted price trends. Figure 4 shows the comparison of Bitcoin closing prices.

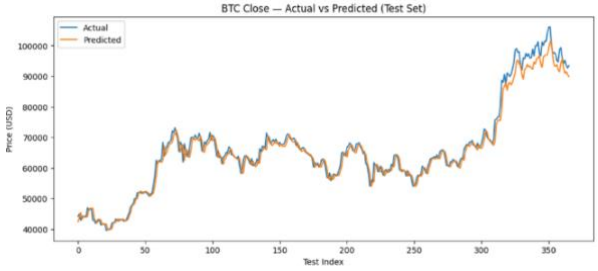


Figure 4. Comparison of Predicted and Actual Bitcoin Closing Prices

After denormalization, the predictions are compared with the actual closing prices in a table containing the date, actual price, CNN-LSTM predictions, and the difference for each observation. This table provides a clearer view of the model's ability to estimate Bitcoin prices in the real market. Table 7 presents the comparison between the predictions and the actual closing prices.

Table 7. Comparison of Predictions after Denormalization

Date	Actual Price (USD)	Predicted Price (USD)	Difference (USD)
2024-12-01	43,200	42,985	215
2024-12-02	42,780	42,950	-170
....	...	...	...

Several types of tests were conducted to evaluate the model's performance, including the calculation of evaluation metrics, model validation through loss curve analysis and TSCV, as well as statistical testing to ensure the reliability of the predictions. Three main evaluation metrics were used: RMSE, MAE, and MAPE. RMSE measures the square root of the average deviation between predicted and actual values. MAE calculates the average absolute difference without considering the direction of the deviation. MAPE expresses the prediction error as a

percentage relative to the actual value, making accuracy easier to interpret. The detailed outcomes are presented in Table 8.

Table 8. Evaluation Metrics Results

Model	RMSE	MAE	MAPE
CNN	1124.32	876.45	4.82%
LSTM	987.56	742.31	3.97%
CNNLSTM	**2,202.71**	**1,553.20**	**2.24%**

From the evaluation results, the single CNN model achieved prediction accuracy with an RMSE of 1124.32, an MAE of 876.45, and MAPE of 4.82%. The single LSTM model performed slightly better with RMSE of 987.56, MAE of 742.31, and MAPE of 3.97%. Meanwhile, the hybrid CNN-LSTM model produced RMSE of 2202.71, MAE of 1553.20, and MAPE of only 2.24%. The RMSE value represents the average prediction error on the same scale as the original data (2,202.71). The MAE of 1,553.20 indicates the average absolute prediction error, while the MAPE of 2.24% reflects the relative error compared to actual values. Based on MAPE, prediction accuracy can be estimated using the formula Accuracy ≈ (100% − MAPE), resulting in 97.76% accuracy. This demonstrates that the model achieves very strong predictive performance with a low relative error rate.

Model validation was also carried out to ensure stability and reliability during training and testing. Two approaches were used: loss curve analysis and TSCV. The loss curve analysis tracked training and validation loss trends across epochs to evaluate convergence, learning stability, and potential overfitting or underfitting. A decreasing and flattening loss curve indicated good learning ability and stable generalization to test data. In addition, TSCV was applied to repeatedly assess performance by splitting the time series data into multiple training and testing subsets in sequence. This approach provides a more accurate evaluation of the model's ability to predict new, sequential, and dynamic data, making the testing results more robust. Through these validation methods, the reliability of the CNN-LSTM model's predictions was confirmed, supporting the conclusion that it is a dependable approach for forecasting Bitcoin's volatile and dynamic closing prices.

The main discussion obtained from the process of implementing and testing the Bitcoin closing price prediction model using the hybrid CNN-LSTM approach is focused on two main aspects, namely hyperparameter optimization and model performance evaluation. In the first part, it explains how the model's hyperparameters were determined through experimental stages, parameter adjustments, and consideration of literature so that the best parameter combination was obtained to support the stability and accuracy of predictions. Furthermore, in the second part, the performance of the hybrid CNN-LSTM model was analyzed by comparing it with single baseline models, namely CNN only and LSTM only, based on the results of error metric calculations and loss curve validation.

From the evaluation results previously presented in Table 8, the single CNN model had the highest error compared to the LSTM and CNN-LSTM models. The CNN model was less optimal in capturing long-term dependencies in time series data because CNN focuses more on local patterns. However, the single CNN model is suitable for cases with short-term price movement patterns but less effective for multi-day predictions. The single LSTM model outperformed CNN because LSTM is specifically designed for sequential or time series data. LSTM is able to learn long-term dependencies in Bitcoin price movements, but it still has shortcomings in capturing local patterns that can be extracted by CNN, and LSTM is slower in training due to its architectural complexity.

The CNN-LSTM model, on the other hand, combines the advantages of both previous models, allowing it to extract local patterns and important features from price data while also processing sequential information to predict long-term trends. Based on the results in Table 8, CNN-LSTM shows the best performance in terms of MAPE, achieving a value of 2.24 percent,

which corresponds to an accuracy of 97.76 percent. However, despite its strong performance in MAPE, the CNN-LSTM model records the highest RMSE and MAE among the three models. This was further demonstrated in Figure 5, which compares the predictions of the single CNN, single LSTM, and CNN-LSTM models with the actual market prices.

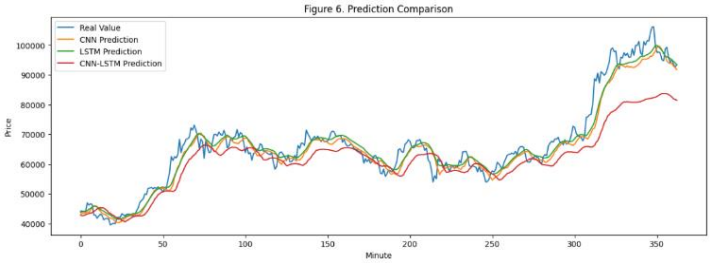


Figure 5. Comparison of Models in Prediction

Figure 5 presents a comparison between the actual Bitcoin closing prices (blue line) and the predictions of three models: the single CNN (orange line), the single LSTM (green line), and the hybrid CNN-LSTM (red line). The CNN prediction tends to follow short-term fluctuations more sharply due to the use of convolutional filters for detecting local patterns, while the LSTM prediction produces a smoother and more stable curve in capturing long-term trends thanks to its sequential memory. The hybrid CNN-LSTM model combines the strengths of both CNN and LSTM, resulting in a curve that is stable in capturing macro trends but slightly dampens extreme price peaks. This pattern indicates that each model has specific advantages: CNN excels in local detail, LSTM in global trend detection, and CNN-LSTM serves as a compromise with stable trend representation but potential underfitting in extreme fluctuations. Figure 6 shows the training loss curves of the three models.

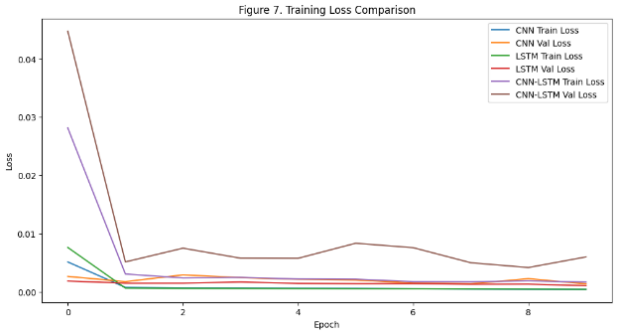


Figure 6. Training Loss Comparison

Figure 6 shows the training and validation loss curves for the single CNN, single LSTM, and hybrid CNN-LSTM models. All models exhibit a sharp loss decrease in the early epochs (1–2), indicating effective initial adaptation to data patterns and weight adjustment. For the CNN and LSTM models, both training loss and validation loss curves tend to stabilize and flatten after the second epoch, indicating that the baseline models converge relatively quickly. The average training loss value for CNN is around 0.005, while the LSTM achieves even lower values, approaching 0.002–0.003 in the final epochs. This aligns with the characteristics of LSTM, which is particularly effective at capturing recurring temporal patterns in time series data.

The hybrid CNN-LSTM model also shows a significant loss reduction in the early epochs, but it records relatively higher training and validation loss values compared to the baseline models during the first epoch. However, despite the higher initial loss, the CNN-LSTM loss trends consistently decrease and stabilize within the range of 0.005–0.007 in subsequent epochs. A small gap between training loss and validation loss in CNN-LSTM indicates that model regularization is functioning, though there may be slight underfitting, especially if dropout or pooling is applied too strongly. Across all three models, no extreme gaps between

training loss and validation loss are observed, which suggests that there are no indications of severe overfitting. This is important, since a wide gap between training and validation loss typically Implies that the model is memorizing training data patterns rather than generalizing.

In CNN and LSTM, the loss curves are nearly parallel and flat, supporting the assumption of good generalization. From these trends, it can be concluded that although the hybrid CNN-LSTM model exhibits slightly higher training loss compared to the baseline models, it still holds advantages by combining the CNN's sensitivity to local patterns with the LSTM's ability to capture long-term memory. This aligns with a study conducted by Omole and Enke (2024) [10]. The difference in loss values can also be explained by the larger number of parameters in the hybrid model, which requires more careful tuning of epochs and batch size during optimization.

The loss curves support the prediction patterns shown previously: CNN-LSTM is relatively more stable in capturing macro trends, albeit slightly suppressing detailed fluctuations. The CNN baseline adapts more quickly to local patterns, while the LSTM baseline achieves the lowest and most stable loss on test data, which supports its strength in modeling long-term time series patterns. Through this discussion, the advantages of the hybrid CNN-LSTM approach are highlighted in recognizing the non-linear and dynamic fluctuations of time series price movements, thereby demonstrating that this model outperforms conventional models in terms of predictive accuracy and generalization capability.

CONCLUSIONS AND RECOMMENDATIONS

The hybrid CNN-LSTM model proved effective in improving the accuracy of Bitcoin closing price predictions compared to conventional and single models such as LSTM. The testing results showed low error values, with MAPE below 5%, confirming that the combination of CNN's strength in extracting local patterns and LSTM's ability to capture temporal dependencies significantly enhanced predictive performance. Data preprocessing steps, including handling missing values, applying min-max normalization, and generating sequential data through a sliding window approach, also played a crucial role in improving input quality and enabling the model to learn historical price patterns more effectively. A comparison of actual and predicted prices demonstrated that the hybrid model was able to follow Bitcoin's price fluctuations with relatively small deviations, while the loss function curve showed a stable convergent trend, indicating no significant overfitting during training. Based on these evaluation metrics, the model shows strong potential for application in decision-making by crypto investors and traders, particularly in anticipating high market volatility and non-linear price movements.

This study was limited to historical Bitcoin price data from Yahoo Finance. Future research may improve accuracy by incorporating other assets such as stocks or indices (e.g., S&P 500, IDX), given the observed correlation between stock and crypto markets. Hyperparameter optimization, including epoch, batch size, and learning rate adjustments, as well as exploring alternative architectures, could further enhance stability and accuracy, particularly during periods of extreme volatility. To support real-time decision-making, future studies are encouraged to develop an interactive dashboard or price prediction bot through a Command Line Interface (CLI) or web application, enabling traders to directly utilize predictions in daily transactions. As external validation, the model can also be tested on other cryptocurrencies such as Ethereum or Litecoin to assess the generalizability of the hybrid CNN-LSTM approach for digital assets with similar volatility characteristics.

REFERENCES

[1] F. Fang *et al.*, "Cryptocurrency Trading: A Comprehensive Survey," *Financ Innov 8*, vol. 13, 2022, doi: https://doi.org/10.1186/s40854-021-00321-6

[2] B. R. Craig and J. Kachovec, "Bitcoin's Decentralized Decision Structure," 2019, doi: https://doi.org/10.26509/frbc-ec-201912
[3] D. Bakas, G. Magkonis, and E. Y. Oh, "What Drives Volatility In Bitcoin Market?," *Financ. Res. Lett.*, vol. 50, 2022, doi: https://doi.org/10.1016/j.frl.2022.103237
[4] S. Guizani and I. K. Nafti, "The Determinants of Bitcoin Price Volatility: An Investigation With Ardl Model," *in Procedia Computer Science, Elsevier B.V.*, vol. 164, pp. 233–238, 2019, doi: https://doi.org/10.1016/j.procs.2019.12.177
[5] Y. Hua, "Bitcoin Price Prediction Using Arima And LSTM," *in E3s Web of Conferences, Edp Sciences*, vol. 218, 2020, doi: https://doi.org/10.1051/e3sconf/202021801050
[6] Y. Li and W. Dai, "Bitcoin Price Forecasting Method Based On Cnn-Lstm Hybrid Neural Network Model," *J. Eng.*, vol. 2020, no. 13, pp. 344–347, 2020, doi: https://doi.org/10.1049/joe.2019.1203
[7] F. P. Rachman and H. Santoso, "Perbandingan Model Deep Learning Untuk Klasifikasi Sentiment Analysis Dengan Teknik Natural Languange Processing," *J. Teknol. Dan Manaj. Inform.*, vol. 7, no. 2, pp. 103–112, 2021, doi: https://doi.org/10.26905/jtmi.v7i2.6506
[8] M. Gholipour, "Leveraging The Power Of Hybrid Models: Combining Arima And LSTM For Accurate Bitcoin Price Forecasting," 2023.
[9] Ferdiansyah, R. Zahilah, S. Hajar, and D. Stiawan, "CNN-LSTM Hybrid Model For Improving Bitcoin Price Prediction Results," *Applied Mathematics and Computational Intelligence (AMCI)*, vol. 12, no. 4, pp. 13–26, 2023, doi: https://doi.org/10.58915/amci.v12i4.349
[10] O. Omole and D. Enke, "Deep Learning For Bitcoin Price Direction Prediction: Models And Trading Strategies Empirically Compared," *Financ. Innov.*, vol. 10, no. 1, 2024, doi: https://doi.org/10.1186/s40854-024-00643-1
[11] S. Somayajulu, M. Ahmed, and B. Kotaiah, "Bitcoin Price Prediction Using LSTM And CNN," 2024.
[12] Q. Guo, S. Lei, Q. Ye, and Z. Fang, "MRC-LSTM: A Hybrid Approach Of Multi-Scale Residual CNN and LSTM To Predict Bitcoin Price," 2021, doi: https://doi.org/10.48550/arXiv.2105.00707
[13] M. Ortu, N. Uras, C. Conversano, G. Destefanis, and S. Bartolucci, "On Technical Trading And Social Media Indicators In Cryptocurrencies' Price Classification Through Deep Learning," *Expert Systems with Applications*, vol. 198, 2022, doi: https://doi.org/10.1016/j.eswa.2022.116804
[14] V. P. Ramadhan and F. Y. Pamuji, "Analisis Perbandingan Algoritma Forecasting Dalam Prediksi Harga Saham LQ45 PT Bank Mandiri Sekuritas (BMRI)," *J. Teknol. dan Manaj. Inform.*, vol. 8, no. 1, pp. 39–45, 2022, doi: https://doi.org/10.26905/jtmi.v8i1.6092
[15] S. Jha and A. Yadav, "Hybrid Deep Learning Model For Bitcoin Price Prediction," *Int. J. Adv. Comput. Sci. Appl.*, Vol. 13, No. 6, 2022.
[16] R. Sharma, A. Singh, and M. Gupta, "Bitcoin Price Prediction Using Hybrid CNN-LSTM Model," *Procedia Comput. Sci.*, vol. 185, 2022.
[17] H. Kim, J. Lee, and S. Park, "Comparative Analysis of Deep Learning Models For Bitcoin Price Prediction," *J. Comput. Sci.*, vol. 50, 2021.
[18] H. Jang and J. Lee, "Modeling And Prediction Of Bitcoin Prices With Bayesian Neural Networks," *IEEE Access*, vol. 8, 2020.
[19] D. Tiwari and A. Kumar, "Attention-Based Cnn-Lstm Model For Cryptocurrency Price Forecasting," *Appl. Soft Comput.*, vol. 123, 2023.
[20] K. A. Rijal, A. V. Vitianingsih, Y. Kristyawan, and A. Lidya, "Forecasting Model Of Indonesia's Oil & Gas And Non-Oil & Gas Export Value Using Var And LSTM Methods," *J. Teknol. dan Manaj. Inform.*, vol. 10, no. 1, pp. 59–69, 2024, doi: https://doi.org/10.26905/jtmi.v10i1.13127