Optimizing Multivariate LSTM Networks for Improved Cryptocurrency Market Analysis

Mohamed O. BEN MILOUD
School of EECS
University of North Dakota
Grand Forks, ND 58202-7165
mohamed.benmiloud@und.edu

Eunjin Kim
School of EECS
University of North Dakota
Grand Forks, ND 58202-7165
ejkim@cs.und.edu

ABSTRACT

Although cryptocurrency has gained popularity as an investment alternative, investors may find it challenging to manage the market due to its high level of volatility. This study focused on Bitcoin and Ethereum over six months and collected and evaluated data on four critical characteristics to assist investors in making more educated decisions. The study discovered that thorough market knowledge is essential and that combining cutting-edge analytical methods like multivariate LSTM networks and Differential Evolution (DE) can improve accuracy and reduce risk in investment decisions. Investors can alter their investments by using data analysis to picture market movements and pinpoint important trends comprehensively. The research represents a significant advancement in the cryptocurrency market analysis and offers insightful information on the variables affecting Bitcoin's market value. It provides a trustworthy tool for investors to make informed investment decisions in a complex and erratic market. Using cutting-edge analytical tools, investors can increase their chances of success in this vibrant and quickly changing market.

Keywords: Cryptocurrency, Bitcoin, Ethereum, Data Analysis, Multivariate LSTM networks, Differential Evolution, Investment decisions

1. INTRODUCTION

Since the launch of the first cryptocurrency Bitcoin by Nakamoto [1] in 2008, the market of cryptocurrencies has developed and grown to contain more than 12,000 cryptocurrencies, especially in the last four years with market capitalization valued at around 1 trillion dollars with Bitcoin representing a dominance of 38%. The market's popularity has grown significantly and has seen a surge in both supply and demand due to the popularity of cryptocurrencies and the influx of additional capital into it. On a wide range of online exchange platforms, cryptocurrency markets are open for trading every day of the week, around-the-clock.

Machine learning has been widely used for cryptocurrency price prediction, with various techniques like neural networks, decision trees, and support vector machines being explored by researchers. Recently, deep learning algorithms, particularly the Long Short-Term Memory (LSTM) model, have demonstrated better accuracy in forecasting cryptocurrency prices. LSTM is a recurrent neural network that can effectively capture long-term dependencies in time-series data, making it suitable for predicting the volatile and non-linear behavior of cryptocurrency prices. As LSTM has been successfully applied in other areas like speech recognition, natural language processing, and image recognition, its promising results in cryptocurrency price prediction have caught the attention of researchers. One of the main advantages of LSTM is its ability to handle multiple variables, including open, high, low, and close prices, volume, and technical indicators like moving averages, Relative Strength Index (RSI), and Moving Average Convergence Divergence (MACD). Incorporating technical indicators in LSTM models has been shown to improve the accuracy of cryptocurrency price prediction [2].

2. BACKGROUND STUDY

Data science tools have made steady progress with encouraging outcomes for economics applications. According to several studies, data science applications to economics can be categorized and researched using various well-known technologies, including deep learning and hybrid learning models. The ability to learn from data and give in-depth insight into issues is provided by machine learning (ML) algorithms. Researchers use machine learning models to address a range of economics-related topics [3]. Current applications of deep learning (DL), a developing discipline of machine learning, include self-driving cars, image identification, hazard prediction, health informatics, economics, and finance [4]. The effectiveness of DL models in comparison to traditional ML models, such as support vector machines (SVM), K-nearest neighbors (KNN), and generalized regression neural networks (GRNN), has been assessed in several comparative studies. DS techniques have advanced quickly, many new industries and disciplines are added to the list of users of DS algorithms. The accuracy of the other models is improved by using hybrid machine learning models, which combine two or more single algorithms. Hybrid models can be created to maximize the prediction function by combining two predictive machine learning algorithms or a machine learning algorithm with an optimization technique.

Similar to investing in the stock market, where the forecast of future value significantly influences investment decisions, investing in cryptocurrencies involves decision making. An

ISBN: 978-1-950492-73-2
ISSN: 2771-0947

https://doi.org/10.54808/WMSCI2023.01.83

intriguing research challenge developing in the literature is using ML and DL models to predict the patterns of cryptocurrency values. Deep learning applications in the stock market have grown more prevalent than in other branches of economics. Financial time series data are noisy and nonstationary, in addition to the stock market's inherent complexity and volatility. To forecast stock values many researchers have used single LSTM or hybrid LSTM [5] [6]. Tamura et al. [7] used technical financial indices of the Japanese stock market. Wang et al. [8] applied the model of LSTM with SVM to portfolio optimization and the prediction of portfolio management's financial time-series and compared the results with the autoregressive integrated moving average model (ARIMA) and deducted that LSTM is more suitable for financial time-series forecasting. Fister et al. [9] designed an automated stock trading system and argues that LSTM performance is more reliable than various traditional trading strategies.

Hybrid LSTM models have been proven to outperform other single deep learning methods in for the long-tern stock price prediction. [10] Shekhar and Varshney [11] integrated a hybrid model with genetic algorithm-SVM (GA-SVM) in order to forecast the direction of the stock market using sentiment analysis. They demonstrated through quantitative empirical study that adding sentiment analysis to GV-SVM model boosted model accuracy by 18.6\%, resulting in a final model with an accuracy of roughly 89.93\%. Ebadati and Mortazavi [12] employed a genetic algorithm to select features and optimize parameters for an ANN. Their study shows that this hybrid model has improved both speed and performance accuracy.

For investors, traders, and experts alike, the unpredictability of the cryptocurrency market has given rise to several worries. The risk associated with cryptocurrency investments also depends on how the system is set up and run. The degree of anonymity offered by the system, any central exchange or central authority controlling the system, long-term system operations, and the price trend will all be of great interest to investors.

3. METHODOLOGY

In this study, we sourced market data on Bitcoin and Ethereum cryptocurrencies from Yahoo Finance API. The data was collected at a daily frequency over a period of six months, from October 20th, 2022, to April 20th, 2023, and comprised of four variables, namely, open price, close price, high price, and low price. To access the data, we employed the widely used programming language Python and the yfinance module, which is an open-source API designed to enable easy and efficient extraction of market data from Yahoo Finance. This approach facilitated a streamlined and efficient data collection process for our analysis.

A unit of the LSTM model is made up of a memory cell and three gates that manage the movement of data into and out of the cell as shown in Figure 1. These gates are the input gate, the forget gate, and the output gate. The input gate regulates the flow of data from the input to the memory cell, while the output gate controls the flow of data from the cell to the output. The forget gate decides which data should be discarded from the memory cell. The basic equations for an LSTM unit are:

Input gate: $i_t = sigmoid(W_i \cdot [h_{t-1}, x_t] + b_i)$ (1)
Forget gate: $f_t = sigmoid(W_f \cdot [h_{t-1}, x_t] + b_f)$ (2)
Output gate: $o_t = sigmoid(W_o \cdot [h_{t-1}, x_t] + b_o)$ (3)
Cell state: $C_t = f_t \cdot C_{t-1} + i_t \cdot tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$ (4)
Hidden state: $h_t = o_t \cdot tanh(C_t)$ (5)

Where i_t, f_t, and o_t are the input, forget, and output gates, respectively, which are sigmoid functions of linear combinations of the current input x_t and the previous hidden state h_{t-1} with learned weights W_i, W_f, and W_o, and biases b_i, b_f, and b_o.
C_t is the cell state at time t, which is a combination of the previous cell state C_{t-1} and a new candidate cell state $i_t \cdot tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$ scaled by the input and forget gates. Here, W_c and b_c are learned weights and biases for the candidate cell state, and tanh is the hyperbolic tangent function.
h_t is the new hidden state at time t, which is a scaled tanh of the current cell state C_t multiplied by the output gate o_t. These equations enable the LSTM network to selectively remember or forget information over time, making it well-suited for processing long sequences of input.

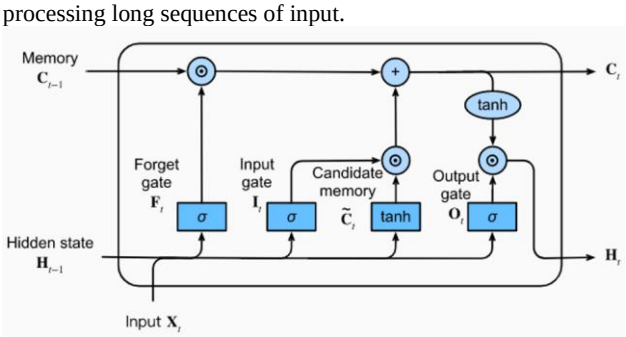

Figure 1 Univariate LSTM Architecture [13]

A variation of the conventional LSTM network called multivariate LSTM (Long Short-Term Memory) is made to accommodate many input features, each of which may have a variable temporal dependence. Each input feature is fed into a separate LSTM cell in a multivariate LSTM, enabling the network to capture the distinct temporal patterns of each feature as shown in Figure 2. The network's final output is created by combining the results of each LSTM cell.
Multivariate LSTMs (m-LSTM) are frequently employed in time series analysis and prediction tasks, such as cryptocurrency price prediction, weather forecasting, and energy consumption forecasting. A multivariate LSTM is able to capture more intricate and nuanced correlations between the variables by utilizing numerous input features, which can result in more accurate predictions and insights.

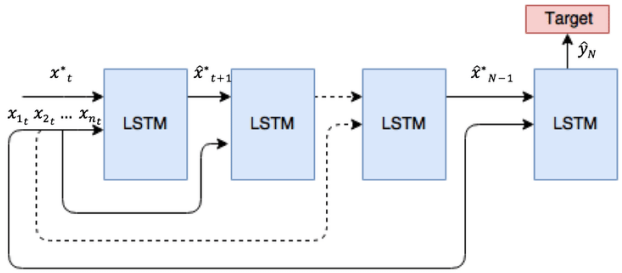

Figure 2 Multivariate LSTM Architecture [14]

Input gate: $i_t = sigmoid(W_i \cdot [h_{t-1}, x_{1,t}, x_{2,t}, ..., x_{n,t}] + b_i)$ (6)
Forget gate; $f_t = sigmoid(W_f \cdot [h_{t-1}, x_{1,t}, x_{2,t}, ..., x_{n,t} + b_f)$ (7)
Output gate: $o_t = sigmoid(W_o \cdot [h_{t-1}, x_{1,t}, x_{2,t}, ..., x_{n,t}] + b_o)$ (8)
Cell state: $C_t = f_t \cdot C_{t-1} + i_t \cdot tanh(W_c \cdot [h_{t-1}, x_{1,t}, x_{2,t}, ..., x_{n,t} + b_c)$ (9)
Hidden state: $h_t = o_t \cdot tanh(C_t)$ (10)

The input vector is denoted by x_t*, where the star symbol (*) is used to indicate that the input vector may be preprocessed or

transformed before being fed into the RNN. This includes techniques such as normalization to prepare input data for machine learning models. the input vector $\hat{x}_t$ is transposed in order to conform to the expected input shape of the LSTM layer. Specifically, an LSTM layer expects its input to be a sequence of vectors, where each vector has a fixed number of dimensions. By transposing the input vector $\hat{x}_t$, the dimensions are rearranged so that $\hat{x}_t$ becomes a column vector with a single column and n rows (where n is the number of dimensions in the input vector). This makes it easier to stack multiple input vectors into a sequence, which can then be fed into the LSTM layer one timestep at a time.

The sole difference between the equations for the univariate LSTM model and for those multivariate-LSTM is the dimension of the input vectors. For example, a cryptocurrency's closing price makes up the input vector in the univariate instance while the multiple attributes of a cryptocurrency such as opening, closing, high, and low prices make up the multivariate input vector. The input, forget, and output gates, the cell state, and the hidden state are the fundamental elements with the same equations for both scenarios. The multivariate-LSTM model can capture the intricate connections between the various features by including all pertinent aspects in the input vector, which enables it to develop a complete knowledge of the behavior of the data. As a result, the multivariate-LSTM is often able to produce more accurate and reliable predictions than the univariate LSTM, especially when dealing with complex time series data.

In contrast to most studies that concentrate on analyzing a single feature, we have opted for multivariate-LSTM (Long Short-Term Memory) to incorporate all the relevant attributes of open, close, high, and low in our analysis. The unique characteristics of the multivariate-LSTM model allow us to apprehend the intricacies of interdependent features, and in turn, comprehend the data's behavior. We anticipate that this innovative approach will yield a more precise and trustworthy data analysis, which is of inestimable value in making sound decisions.

The LSTM has its own set of restrictions and potential issues, much like any other machine learning model. The following are a few LSTM-related problems that could occur:

- Overfitting: LSTM models can occasionally be overfitted to the training data, which results in their poor performance on new data. Too many parameters or insufficient regularization are two examples of issues that can contribute to this.
- LSTMs may not be the ideal option for all sorts of data and may perform poorly on data that is highly non-linear, has long-term dependencies, or has other problematic qualities.
- Difficulty in finding optimal hyperparameters: There are many hyperparameters that can be adjusted in an LSTM model, such as the number of hidden layers, the number of neurons per layer, and the learning rate. Finding the optimal values for these hyperparameters can be a difficult and time-consuming task.

By improving the input data and hyperparameters of an m-LSTM model, differential evolution can be utilized to alleviate some of these problems. The model's performance can be enhanced, and the likelihood of overfitting can be decreased by utilizing differential evolution to find the ideal set of input features and hyperparameters. Due to its simplicity, effectiveness, and robustness, Differential Evolution (DE) is a potent optimization technique that has been extensively applied in numerous academic domains. DE is a population-based stochastic algorithm that simulates the natural process of evolution. A population of potential solutions is iteratively improved through the processes of selection, recombination, and mutation in DE.

Due to DE's aptitude for dealing with high-dimensional issues, ability to converge to global optima, and capacity for overcoming local optima, our multivariate-LSTM model will be optimized using DE in this work. Many studies demonstrated that DE outperforms other optimization methods in terms of accuracy and efficiency when it comes to optimizing the hyperparameters of deep learning models, including LSTMs [15].

The m-LSTM model needs to be trained after its architecture has been established. The number of m-LSTM units, layers, and output dimensions are just a few examples of the hyperparameters of the model that are optimized using DE. The population is assessed by computing the fitness of each member of the population at the end of each generation after the algorithm has run for a predetermined number of generations.

The performance of optimized m-LSTM on the testing set is assessed after it has been trained using the current population of hyperparameters to derive the fitness function. The fitness score is determined using a statistic that gauges how accurately the model predicts the future. The fitness ratings are then used by the differential evolution algorithm to create a new population of hyperparameters for the following generation. Up until the predetermined number of generations has been attained, this process is repeated.

Figure 3 outlines the steps involved in building a trained m-LSTM model for predicting opening prices. The input consists of multivariate input data (open, close, high, low) and univariate target data (opening price), as well as various hyperparameters including population size, differential evolution factor, crossover probability, number of training epochs, m-LSTM layer units, dropout rate, learning rate, and batch size.

The m-LSTM model is defined by initializing a sequential model, adding an m-LSTM layer with the specified number of units, a dropout layer, and a dense layer for predicting opening price. The fitness function evaluates the mean squared error (MSE) of the model on the validation set, and the differential evolution algorithm is used to find the best set of hyperparameters. The trained m-LSTM model is returned as output and can be used to make predictions on new input data using the predict() method.

To select the optimal hyperparameters for a neural network model using differential evolution, we first define the search space for each hyperparameter. We consider the following hyperparameters: the number of neurons in the input layer, the number of neurons in each of the two hidden layers, the number of output dimensions, the number of epochs, and the batch size.

The number of units in a neural network might vary in practical applications depending on the task at hand. Depending on the difficulty of the problem and the quantity of available data, the number of units is often set between 10 and 1000.

We have chosen to restrict the number of units for the differential method to a range between 128 and 512. This choice was made considering prior research demonstrating that this range of units is efficient for a variety of tasks and strikes a reasonable balance between model complexity and training duration.

Algorithm 1 Pseudocode of the suggested optimization method

1) Define the architecture of the m-LSTM model, including the number of units, dropout rate, and input shape.
2) Define the fitness function, which will be used to evaluate the performance of the model.
3) Optimization of the hyperparameters of mLSTM by DE
   - 3.1 Initialize a population of candidate models, each with a unique set of hyperparameters.
   - 3.2 For each candidate model, calculate its fitness score by training it on a validation set and evaluating its performance using the fitness function.
   - 3.3 Select three other candidate models at random, and use them to create a new candidate model through mutation and crossover.
   - 3.4 Evaluate the fitness score of the new candidate model, and replace the original candidate model with it if it has a better fitness score.
   - 3.5 Repeat steps 3.2 – 3.4 for a maximum number of generations or until a satisfactory.
4) Train the best candidate model of m-LSTM using the training dataset.
5) Use the trained m-LSTM model to make predictions on new data.

Figure 3 Pseudocode of the suggested hybrid model

The range of values to be used for the differential evolution algorithm's epochs and batch size relies on a number of variables, including the size and complexity of the dataset, the design of the neural network, and the available processing resources.

A fair starting point for the number of epochs could be a range of 10 to 1000, with increments of 10. However, if the dataset is modest, fewer epochs can be required. On the other hand, a larger number of epochs can be required to obtain better performance if the dataset is huge and complex.

An acceptable range for the batch size could be between 16 and 128, in increments of 16. Again, the size of the dataset and the available computer resources will determine the ideal batch size. Faster convergence can be achieved by updating the model weights more frequently with a lower batch size, but this comes at the expense of longer computation times. On the other hand, better gradient estimates can be obtained with a larger batch size, but at the expense of slower convergence.

4. RESULTS

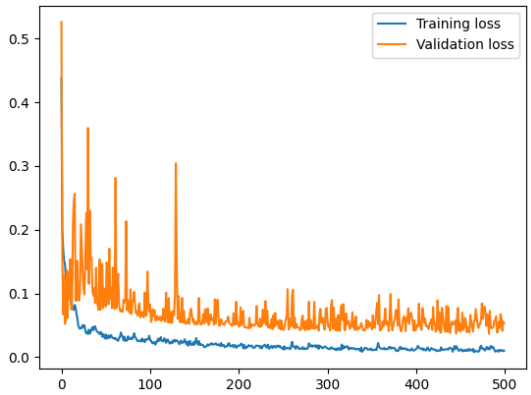

Figure 4 TL & VL plot before DE optimization for Bitcoin

The results presented depict the initial outcomes of the Multivariate LSTM network in contrast to the results obtained post application of the optimization recommended by the differential evolution algorithm, which applied to Bitcoin and Ethereum.

The figures presented in the study visually represent the model's performance for Bitcoin and Ethereum. Figures 4 and 5 illustrate the performance of the Multivariate LSTM network for Bitcoin using the metrics of Training Loss and Validation Loss, respectively. The model's performance improved significantly after hyperparameter optimization, as shown in Figure 5. Figures 5 and 6 display the optimized model's performance for Ethereum, indicating a notable improvement in the model's performance. Furthermore, the significant reduction in error after optimization is evident in Figure 7.

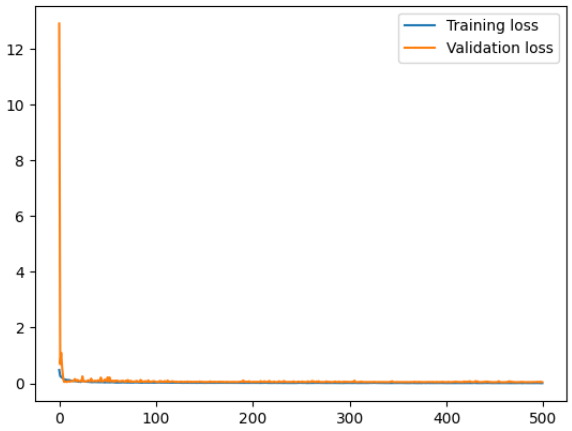

Figure 5TL & VL plot after DE optimization for Bitcoin

To evaluate the effectiveness of the differential evolution optimization algorithm in improving the performance of the LSTM models for both Bitcoin and Ethereum, Tables 1 and 2 were generated to present the performance metrics of the initial (unoptimized) model compared to the optimized model. These tables serve as a comparative analysis between the two models, highlighting the improvements achieved through the optimization process. Specifically, the tables provide an overview of the different performance metrics, such as test loss, mean absolute error (MAE), mean squared error (MSE), mean absolute percentage error (MAPE), and cosine similarity, before and after optimization.

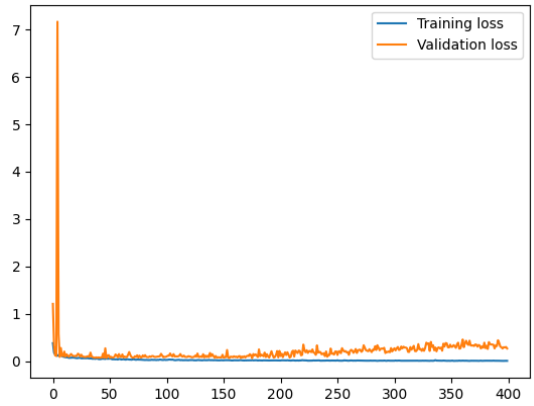

Figure 6 TL & VL plot before DE optimization for Ethereum

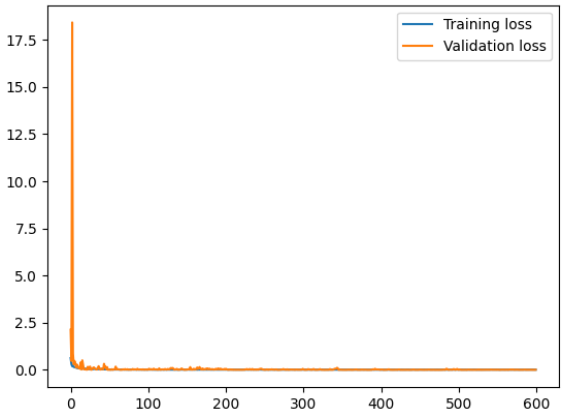

Figure 7 Optimized model application to Ethereum (TL & VL)

Table 1 Performance metrics of Bitcoin analysis

Performance metrics	Test Loss	Test MAE	Test MSE	Test MAPE	Test Cosine Similarity
Initial model	0.0518	0.1445	0.0426	48.3205	0.9754
Optimized model	0.0016	0.0328	0.0016	6.4330	1.0

Table 2 Performance metrics of Ethereum analysis

Performance metrics	Test Loss	Test MAE	Test MSE	Test MAPE	Test Cosine Similarity
Initial model	0.0426	0.1469	0.0518	208.344	0.9141
Optimized model	0.0034	0.0489	0.0034	7.8112	0.9881

5. DISCUSSION

Regarding the results for the Multivariate LSTM network applied to Bitcoin, it was observed that while the initial results were close to the actual price, they were not accurate. It indicates that the model is overfitting to the training data, meaning it is too complex and cannot generalize well to new data. On the other hand, if the validation loss is much lower than the training loss, it suggests that the model is underfitting, meaning it is too simple and cannot capture the underlying patterns in the data. However, after the optimization suggested by the differential evolution algorithm, a notable improvement was observed in Bitcoin's training loss and validation loss. When the training loss and validation loss are getting closer, it indicates that the model is becoming more balanced and generalizes well to new data This indicates that the model could predict Bitcoin's future value better, reducing the risk for investors.

Similarly, Ethereum's initial results for the Multivariate LSTM network were close to the actual price but not entirely accurate. However, after applying the optimization suggested by the differential evolution algorithm, the training loss and validation loss for Ethereum showed improvement. This infers that using advanced analytical tools like multivariate LSTM networks and Differential Evolution (DE) can enhance the model's accuracy and provide valuable insights for investors.

The results presented show the performance improvement of an m-LSTM model applied to Bitcoin and Ethereum analysis using Differential Evolution optimization. The improved performance of the model with the optimization technique can be attributed to differential evolution being a robust metaheuristic optimization algorithm that can effectively search for optimal hyperparameters of the m-LSTM model.

The experiment shows that the optimized m-LSTM model significantly outperforms the baseline model. Specifically, the optimized model achieves a lower mean squared error (MSE) than the baseline model for both Bitcoin and Ethereum. This performance improvement is consistent with previous research on applying metaheuristic optimization algorithms to machine learning models.

The improved model performance with the optimization technique has several practical implications. Firstly, it can be used by traders and investors looking for accurate predictions of Bitcoin and Ethereum prices. Accurate predictions can help them make informed decisions and increase their chances of making profitable trades. Secondly, the optimized m-LSTM model can be used by researchers interested in studying the behavior of Bitcoin and Ethereum prices. Accurate predictions can help them develop better models and understand the factors that influence the costs of these cryptocurrencies.

However, it is essential to note that there are limitations to the study. For instance, the results are based on historical data and may not reflect the current market conditions. Additionally, the study does not account for external factors that may influence the prices of Bitcoin and Ethereum, such as regulatory changes or technological advancements.

Table 1 presents the performance metrics for Bitcoin before and after optimization. Before optimization, the model's test loss, MAE, and MSE were all relatively high, indicating that its predictions were not very accurate. The model's test MAPE of 48.32% suggested that the average percentage error in its predictions was relatively high. After optimization, the model's low test mean absolute error (MAE) and mean squared error (MSE) scores indicated accurate predictions with relatively small errors. The low value of test loss was also a good indication of the model's performance. The model's common test MAPE suggested consistently accurate predictions, and the perfect cosine similarity score of 1.0 indicated that the predicted values were very similar to the actual values. Overall, the results suggested that the model performed very well on the test set.

For Ethereum, as presented in Table 2, the model's test loss and MSE were moderately low before optimization, indicating reasonably good performance. However, the relatively high test MAE suggested that the model's predictions may be off by an average of 0.1469 units. The high test MAPE of 208.3% indicated a significant deviation from the actual values. The cosine similarity score of 0.9141 suggested that the model had captured some patterns in the data but may not have learned all the underlying relationships between the input and output variables. After optimization, the model's low test loss, MAE, and MSE scores suggested accurate predictions. The MAPE score indicated that the model's predictions were within 7.81% of the actual values, on average. The cosine similarity score of 0.988 indicated that the model's predictions were similar to the actual values. Overall, the results suggested that the model was

performing well and making accurate predictions on the test data for Ethereum.

The study's results highlight the potential benefits of using advanced analytical tools like multivariate LSTM networks and Differential Evolution (DE) for accurate analysis of cryptocurrencies like Bitcoin and Ethereum. While the initial results were close, they were not highly accurate, and the optimization suggested by the differential evolution algorithm helped improve the model's accuracy. These findings can help investors make more informed decisions and reduce risks in the volatile cryptocurrency market.

6. CONCLUSION

In conclusion, the study shows that advanced analytical tools like multivariate LSTM networks and Differential Evolution (DE) can enhance the accuracy of cryptocurrencies analysis like Bitcoin and Ethereum. The initial results were not highly accurate, indicating that the model was overfitting or underfitting the data. However, after applying the optimization by the Differential Evolution algorithm, there was a notable improvement in the training loss and validation loss for both cryptocurrencies. This indicates that the model can now better generalize to new data, which could reduce the risks for investors in the volatile cryptocurrency market. Therefore, using advanced analytical tools like multivariate-LSTM networks with Differential Evolution (DE) can provide valuable insights for investors in making more informed decisions.

The study's findings offer valuable insights into the potential of applying differential evolution optimization to enhance the predictive performance of m-LSTM models in various time series problems. The optimized models could be beneficial for traders, investors, and researchers who seek to understand and predict the behavior of different types of time series data, not just limited to cryptocurrency prices.

7. REFERENCES

[1] S. Nakamoto, "**Bitcoin: A Peer-to-Peer Electronic Cash System**," *Decentralized business review*, p. 21260, 2008.

[2] M. A. Stefano Cavalli, "**CNN-based multivariate data analysis for bitcoin trend prediction**," *Elsevier - Applied Soft Computing*, vol. 101, p. 107065, 2021.

[3] S. a. M. A. a. K. R. a. A. S. a. A. F. Nosratabadi, **State of the art survey of deep learning and machine learning models for smart cities and urban sustainability**, Springer, 2020, pp. 228--238.

[4] P. a. P. T. Gogas, "**Machine learning in economics and finance**," *Computational Economics*, vol. 57, no. Springer, pp. 1--4, 2021.

[5] M. a. H. K. Kyoung-Sook, "**Performance of deep learning in prediction of stock market volatility**," *Economic Computation & Economic Cybernetics Studies & Research*, vol. 53, 2019.

[6] T. a. K. C. Fischer, "**Deep learning with long short-term memory networks for financial market predictions**," *Elsevier - European journal of operational research*, vol. 270, pp. 654-669, 2018.

[7] K. a. U. K. a. I. S. a. M. Y. Tamura, "**Model for evaluation of stock values by ensemble model using deep learning**," *ransactions of the Japanese Society for Artificial Intelligence*, 2018.

[8] W. a. L. W. a. Z. N. a. L. K. Wang, "**Portfolio formation with preselection using deep learning from long-term financial data**," *Expert Systems with Applications*, vol. 143, no. Elsevier, p. 113042, 2020.

[9] D. a. M. J. a. J. Fister, "**Deep learning for stock market trading: a superior trading strategy**," *Neural Network World - Czech Technical University in Prague, Faculty of Transportation Sciences*, vol. 29, pp. 151-171, 2019.

[10] C. a. P. W. a. S. R. a. S. A. Stoean, "**Deep architectures for long-term stock price prediction with a heuristic-based strategy for trading simulations**," *PloS one*, vol. 14, no. Public Library of Science San Francisco, CA USA, p. e0223593, 2019.

[11] S. a. V. N. Shekhar, "**A hybrid GA-SVM and sentiment analysis for forecasting stock market movement direction**," *Test Eng Manage*, vol. 82, pp. 64--72, 2020.

[12] O. a. M. M. Ebadati, "**An efficient hybrid machine learning method for time series stock market forecasting**," *Neural Network World*, vol. 28, no. 10.14311/NNW.2018.28.003, pp. 41-55, 2018.

[13] O. Calzone, "**An Intuitive Explanation of LSTM**," Medium, 21 02 2022. [Online]. Available: https://medium.com/@ottaviocalzone/an-intuitive-explanation-of-lstm-a035eb6ab42c. [Accessed 20 4 2023].

[14] T. w. H. C. B. Aya Ismail, "**Improving Long-Horizon Forecasts with Expectation-Biased LSTM Networks**," arXiv:1804.06776, 2018.

[15] J. a. G. S. a. B. B. a. M. M. a. Z. V. Brest, "**Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems**," *IEEE transactions on evolutionary computation*, vol. 10, no. IEEE, pp. 646--657, 2006.