JOIV : Int. J. Inform. Visualization, 6(2) - June 2022 356-363

INTERNATIONAL JOURNAL ON INFORMATICS VISUALIZATION

journal homepage : www.joiv.org/index.php/joiv

MLP-NARX Bitcoin Price Prediction Model Integrating System Identification Modelling Principles

Muhammad Nazrin Farhan Nasarudin [a], Ihsan Mohd Yassin [b,*], Megat Syahirul Amin Megat Ali [b], Mohd Khairil Adzhar Mahmood [b], Rahimi Baharom [a], Zairi Ismael Rizman [c]

[a] *School of Electrical Engineering, College of Engineering, Universiti Teknologi MARA, 40450 Shah Alam, Selangor, Malaysia*
[b] *Microwave Research Institute, Universiti Teknologi MARA, 40450 Shah Alam, Selangor, Malaysia*
[c] *School of Electrical Engineering, College of Engineering, Universiti Teknologi MARA, 23000 Dungun, Terengganu, Malaysia*
*Corresponding author: *ihsan.yassin@gmail.com*

***Abstract*— Bitcoin is a decentralized digital currency that enables people to exchange value without requiring a third-party intermediary. Due to its many advantages, it has received much interest from institutional and individual investors. Despite its meteoric increase, the price of Bitcoin is an extremely volatile asset class as it purely relies on supply and demand. This presents an interesting opportunity to create a forecasting model. However, many research papers in this area do not analyze the residuals as part of the forecasting resulting in potentially biased models. In this paper, we demonstrate System Identification (SI) residual analysis techniques for the analysis of our forecasting model. The Multi-Layer Perceptron (MLP) Nonlinear Autoregressive with Exogeneous Inputs (NARX) uses historical price data and several technical indicators to predict the future price movements of Bitcoin. The Particle Swarm Optimization (PSO) algorithm was used to find optimal parameters for the model. The model was able to predict one-day price in the prediction test. The model has successfully captured the dynamics of the data through the tests performed on residuals. It also proves the randomness of residuals, albeit with some minor violations.**

***Keywords*— Forecasting; modeling; bitcoin; system identification; artificial intelligence.**

Manuscript received 20 Dec. 2021; revised 10 Feb. 2022; accepted 5 Apr. 2022. Date of publication 30 Jun. 2022.

I. INTRODUCTION

First introduced in 1991 by individuals or individuals working under the pseudonym Satoshi Nakamoto, Bitcoin is a decentralized digital currency that enables people to exchange value without requiring a third-party intermediary [1], [2]. The blockchain technology behind Bitcoin is a publicly distributed ledger containing a series of data blocks linked together and protected using cryptographic hash (mathematically calculated digital fingerprints unique to the data). Any changes made to the data alter the hash. Because the previous block's hash is included in the new block, any changes to the blockchain can be detected and rejected through peer-to-peer consensus of specialized computers that store a copy of the blockchain. Among Bitcoin's advantages are anonymity, security, and traceability [3]–[6].

Investors and speculators around the world trade more than 6,000 cryptocurrencies with a value of USD 2.3 trillion in the relatively young and constantly open cryptocurrency market [7], [8]. Due to the absence of agreed-upon fundamentals to back up Bitcoin's price and pure supply and demand nature [9], [10], its valuation has often been described as highly volatile and uncertain [5], [7], [11]–[14]. The price is highly influenced by sentiment and exposure to the public and traditional/social media coverage [15], [16].

Financial market forecasting methods can be divided into fundamental (examination of past performance causes for price movement) and technical analysis (examination of price movements purely on its market value and historical standpoint) [10], [17]. Furthermore, there are two distinct schools of thought in forecasting financial markets in data representation. The economist model argues that market data is linear, while soft computing methods prefer non-linear models to represent data considered to be non-linear, complex, and dynamic [18]. The increase in popularity of Bitcoin and other cryptocurrencies, together with their unusual nature of the market and potential profits, has led to much research interest [6], [11], [16]. Two main approaches emerge: one tries to characterize the pricing model based on

sentiment [15], [16], while others try to model the price based on fundamental and technical analysis.

AI-based methods tend to treat the data as non-linear and combine both fundamental and technical analysis in their predictions [10]. Additionally, these methods have been proven versatile and successful in problems with unknown or chaotic patterns and high noise as they relax the requirements of formal definition or probability distribution of the data [10], [12]. Among the AI-based methods, RNN and LSTM have been predominantly used in Bitcoin price prediction. RNNs (such as LSTM) excel in time-series forecasting due to their ability to mimic biological memory and efficiently capture relationships between temporal-related data [14]. This network class combines previous and present input/output data as inputs to predict future values. This is done through a series of unique gates that control information flow in the network [17].

LSTM [17] was used to forecast Bitcoin price movements based on order book data. 160,000 data were separated into 80:20 training ratios and used to train the LSTM to output three price predictions: up, down, or stationary. Trading strategies based on LSTM predictions were simulated with modest returns. In Barry and Crane [13], motifs were used to assist an LSTM network in predicting Bitcoin prices. Motifs are repeating sub-sequences present in a time-series dataset. The patterns were used to set the initial conditions for LSTM training. The method suggests that the use of motifs to initialize LSTM networks greatly improved prediction accuracy (reduction of 8% RMSE). A comparison between CNN and LSTM for Bitcoin price prediction was performed by Misnik et al. [14]. The CNN method was unique as it used price chart images as the input. The LSTM was trained with five inputs (price, coefficient of relative change, and three harmonic component ranges). LSTM was found to be superior to CNN, with a statistically significant F-test value.

Other various statistical and AI-based methods reported were GARCH [9], Random Forest (RF) [7], Decision Tree Regressor (DTR) [11], and KNN [11]. Bitcoin mining information was used to predict Bitcoin returns [9]. Information such as hash rate, mining difficulty, and block size to determine future Bitcoin prices. Simulations on the GARCH-M model suggest that mining difficulty and block size were inversely proportionate to Bitcoin price. In Alahmari [11], the Decision Tree Regressor and KNN non-linear algorithms were used to forecast the closing prices of Bitcoin, Ripple, and Ethereum. Hourly data were collected over a span of five to seven years, divided into an 80:20 percent training-testing ratio, and modeled using DTR and KNN. DTR was selected as the best-performing algorithm with above 90% R-squared fit. Avoird [7] used a combination of technical indicators and historical prices as features for an RF-based price predictor. The TI hyperparameters were optimized using Random Search in AutoML. Simulated trading was done over short and long-time intervals. The algorithm yielded 54.54% - 60.23% accuracy. Similarly, Indera et al. [19] used moving average indicators to assist a NARX model predictor of Bitcoin price. NARX parameters, such as input lag, output lag, hidden layer size, and Mersenne-Twister (MT) random seed, were optimized using PSO. Avoird [7] and Indera et al. [19] have used technical indicators to improve prediction accuracy.

Sentiment-based approaches have also been reported. Yang [16] analyzed the correlation of Bitcoin price to sentiment based on news media frequency. Related articles from two major newspapers were analyzed and compared to Bitcoin prices at the time of publication. The research discovered that any coverage (regardless of positive or negative news) related to Bitcoin positively influenced investment returns, with positive news having twice the impact compared to negative ones. Similarly, McCoy and Rahimi [15] discovered Thanews, and Twitter volume is highly correlated with Bitcoin market sentiment, and these sources can yield considerable profits if included as a trading strategy.

A. Problem Statement and Proposed Work

Salient points from the literature suggest that Bitcoin price prediction has been extensively studied and modeled. However, many of these works did not examine model bias in their prediction works (see [6], [7], [11]–[14], [20] for examples). Regression models should rely on residuals analysis to check model assumptions and prevent deviations. Model bias occurs when the predictor does not fully represent the behavior of the system under study. When the difference between the actual and predicted system is compared, the residuals are not stochastic (random) in nature. The theory behind this is that if a model is deterministic, then its predictive power (coefficients) should fully explain the dependent variable in the model, leaving only noise behind. However, if the residuals are not consistent with random error, the dependence of the variable should not be significant, and the model's regression coefficients cannot be trusted.

In this paper, we apply the fundamentals of System Identification (SI), a control engineering discipline to the Bitcoin price prediction problem. SI is concerned with representing systems in mathematical models. Methods in SI emphasize the randomness of the residuals as part of model validation to ensure that the model is valid and acceptable. The model selected is the MLP-NARX model, a shallow neural network structured to accept recurrent connections from past inputs and outputs to assist in the prediction of future outputs.

Additionally, we extend the works of Indera et al. [19] to include several common technical indicators (Moving Average (MA), Relative Strength Index (RSI), and Bollinger Bands (BB)) as input to the MLP-NARX to assist in the prediction. Another contribution is the use of PSO to optimize the MLP-NARX parameters, such as input lag, output lag, hidden layer size, MT random seed, and the technical indicator windows. We believe that the automated methods presented here would greatly simplify the search for the best prediction model for this task.

II. Material and Method

This research presents the structure of experiments carried out in this paper. It begins with a description of the hardware and software used, followed by a description of the experiments. The SI model validation is divided into model fit and residual tests. In Figure 1, the corresponding objectives pertaining to the experiments are also highlighted.

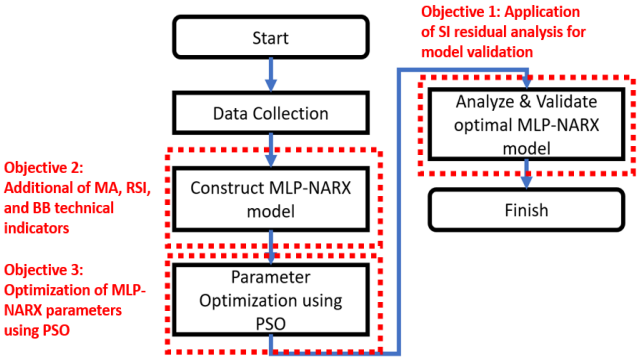

Fig. 1 Experiment flowchart

A. Hardware and Software Used

The computer specifications used for this experiment are shown in Table 1.

TABLE I
HARDWARE AND SOFTWARE SPECIFICATIONS

Item	Specification
Central Processing Unit (CPU)	AMD Threadripper 3990X
Random Access Memory (RAM)	64 GB DDR4 RAM
GPU	NVidia GTX1080 Ti
MATLAB Version	Release 2021a

B. Experiment Description

The experiment begins with data collection and construction of technical indicators and the MLP-NARX model, followed by parameter optimization using PSO and model validation.

1) Data Collection and Preparation

Daily historical Bitcoin prices from Apr. 30, 2016, to Apr. 03 2021 were obtained from CoinDesk data [21]. The data consisted of five elements: opening, low, high, closing prices, and trading volume, as shown in Fig. 2 and Fig. 3, respectively. The data was pre-processed to remove missing values, normalize, and scale the magnitudes. Daily records missing any of the five elements were removed. Normalization was performed to remove trends. It was done by transforming the data such that the mean and standard was 0 and 1, respectively. This step eliminates distribution assumptions of the data, thus improving the result of the MLP-NARX model. Finally, magnitude scaling was used to scale the data to between -1 and +1. This would help the MLP-NARX to learn better since all inputs are within the activation region of the tangent-sigmoid activation function used in the hidden layer.

2) Generate Technical Indicators

Technical indicators are mathematical and statistical techniques that attempt to forecast an asset's short-term future price. They are commonly used by day traders that rely on technical analysis to make trading decisions. Apart from the five inputs listed, three technical indicators - Moving Average (MA), Bollinger Bands (BB), and relative strength index (RSI) were added as inputs to the MLP-NARX model to help improve its prediction ability. All three indicators have modifiable parameters that influence the indicators' values, and they were modified using the PSO algorithm.

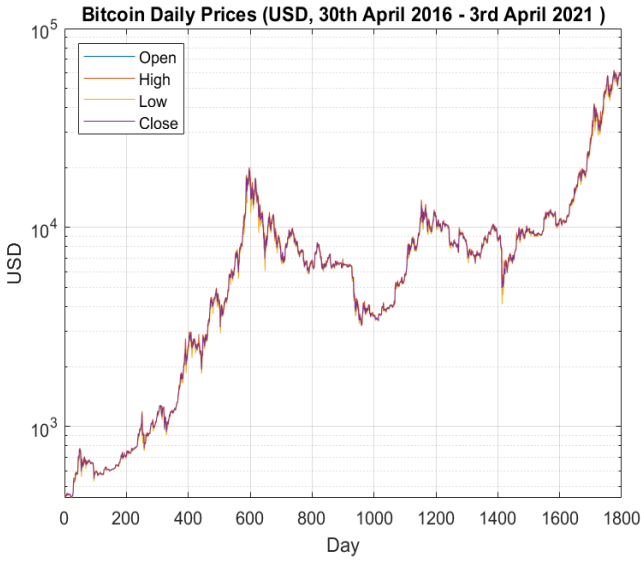

Fig. 2 Bitcoin daily prices from Apr. 30 2016 to Apr. 03 2021

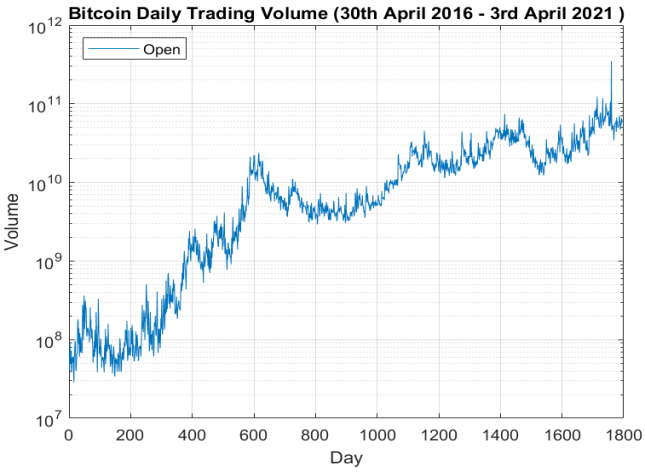

Fig. 3 Bitcoin daily volume from Apr. 30 2016 to Apr. 03 2021

MA is a trend discovery method that uses a series of averages of distinct subsets of the whole data set to examine data points [22]. MA averages data points at specific time intervals in a moving window throughout the dataset. The averaging process helps to smoothen out noise so that clear trends can be discovered. MA can also be used as a gauge for resistance and support levels when the market is heading upwards or downwards, respectively. With the input from this indicator, the prediction model will know the limitation of the price fluctuation when it on uptrend or downtrend.

BB is typically used to discover the beginning and end of trends. It is plotted at a standard deviation level above (upper band) and below (lower band) a simple moving average of the price [23]. Since Bitcoin is dependent on law of supply and demand, BB indicates whether Bitcoin is either oversold or overbought at any point in time. Overbought condition occurs when the price is close to the upper BB, while oversold occurs when the price is close to the lower BB. Like BB, RSI is a momentum indicator that evaluates measures overbought or oversold conditions for an asset based on the magnitude of recent price changes.

3) Construct MLP-NARX model

The Nonlinear Auto-Regressive Moving Average Model with Exogenous Inputs (NARMAX) model is widely considered to be a unified and accurate representation of a wide variety of prediction models [24]. The NARMAX model has several derivatives, including NARX, a non-linear version of the ARX model [20] that performs well in non-linear dynamic systems. The model is represented by Eq. (1).

$$y(t) = f^d \begin{bmatrix} y(t-1), y(t-2), \dots, y(t-n_y), \\ u(t-n_k), u(t-2), \dots, u(t-n_u) \end{bmatrix} + \varepsilon(t) \quad (1)$$

where f^d is the estimated model, $y(t-1), y(t-2), \dots, y(t-n_y)$ are lagged output terms, and $u(t-n_k), u(t-2), \dots, u(t-n_u)$ are lagged input terms, Parameter n_k is the input signal time delay, its value is usually 1 except for cases where the input $u(t)$ is required for identification (in which case, $n_k = 0$) [25].

The model f^d can be created using various statistical or AI-based algorithms. The MLP neural network was used for this research to create the NARX model (Fig. 4) since it is an established method with a proven track record. The MLP-NARX model was configured as a three-layer network function approximation network with a tangent-sigmoid activation function at its hidden layer and linear activation function at the output layer. The MLP-NARX network has several optimizable parameters – input (n_u) and output (n_y) lags, and the number of its hidden units (h). These values were also optimized using the PSO algorithm.

The NARX network trains the network in an open-loop series-parallel architecture. Levenberg-Marquardt (LM) was chosen as the training algorithm due to its proven performance for function approximation networks. Nguyen-Widrow (NW) was used to initialize the weights [26]. It randomly distributes the initial weights between active regions in the hidden and output activation functions to accelerate training. Additionally, Early Stopping (ES) was used to prevent overfitting. ES divides the dataset into training, validation, and testing datasets according to a predefined ratio (in our case, the ratio was 70:15:15). The training dataset was actively used to adjust the MLP-NARX weights, while the validation set monitors training for signs of overfitting. The training was stopped when this occurred. The testing set is an unbiased and independent dataset used to examine the performance of the network post-training.

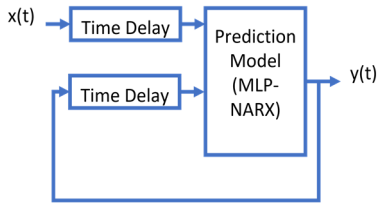

Fig. 4 Structure of MLP-NARX model

4) Parameter optimization using PSO

PSO is a population-based optimization method inspired by the organizational behavior of swarming animals in search of food. The movement of the particles (members of the swarm) is guided by their personal best position (p_{best}) as well as the swarm's best position (g_{best}). The particle's direction, its personal best record, and the swarm's best record are then updated after each iteration until the terminating condition is reached (either the objective is achieved, or the objective has been discovered. PSO is a memory and computationally efficient stochastic optimization algorithm due to its simple mathematical operations and requires minimum memory [27]. The guiding equations of the PSO algorithm are shown in Eq. (2) and Eq. (3).

$$v_{id} = \mathsf{C}\begin{bmatrix} v_{id} + c_1 \times r_1 \times (p_{best} - x_{id}) + \\ c_2 \times r_2 \times (g_{best} - x_{id}) \end{bmatrix} \quad (2)$$

$$x_{id} = x_{id} + v_{id} \quad (3)$$

where C is the constriction factor (Eq. 4):

$$\mathsf{C} = \frac{2}{\left|2-\varphi-\sqrt{\varphi^2-4\varphi}\right|} \ni \varphi = c_1 + c_2 \; and \; \varphi > 4 \quad (4)$$

Altogether, the MLP-NARX inputs involve the adjustment of seven parameters, namely the number of MLP hidden units (h), input lag (n_u), output lag (n_y) and the random seed (r). Additionally, technical indicators used also involve the adjustment of window sizes for each (w_{MA}, w_{RSI}, w_{BB}). These windows were also adjusted automatically using PSO. The fitness function minimizes the cumulative Mean Squared Error (MSE) of the training, validation, and testing sets.

The PSO algorithm itself has several main modifiable parameters - swarm size, the number of particles, and the random seed. The swarm size and particle values should be sufficiently large to encompass some meaningful part of the solution space. However, the values must not be too large that it consumes too much computational power. The random seed value is a pointer to the numbers generated by the Mersenne-Twister (MT) pseudo-random number generator in MATLAB. The seed will set the initial particle position at the same position before optimization.

Secondary PSO parameters control how the particles traverse the solution space. They are shown in Table 2. The velocity range (v_{min} and y_{max}) determines the minimum and maximum values the particles can move from their current position at each iteration. The range of positions (x_{min} and x_{max}) ensures that the solutions stay away from unfeasible regions. The v_{min} and y_{max} values were set to between -1 and +1, while the x_{min} and x_{max} values were set to between 0 and +1. These settings ensure that the solution area is always within 0 and 1.

The values were then subjected to a scaling equation to transform them into the desired individual range of the parameter values (Eq. 5):

$$a = \frac{(a_{max}-a_{min})(b-b_{min})}{b_{max}-b_{min}} + a_{min} \quad (5)$$

where:

a = scaled value
b = original value
a_{min} = desired minimum target range
a_{max} = desired maximum target range
b_{min} = minimum original input range
b_{max} = maximum original input range

TABLE II
PSO PARAMETER VALUES

Parameter	Value
Swarm Size	50
Maximum Iteration	100
Random Seed, r	0, 10,000, 20,000
v_{min}	-1
v_{max}	+1
x_{min}	0
x_{max}	+1
n_u range	1 to 30
n_y range	1 to 30
h	1 to 30
w_{MA}, w_{BB}, w_{RSI} range	1 to 30

C. Model Validation

Model validation is necessary to guarantee that the model appropriately describes the system's behavior. The tests for model validation are categorized into two – model fit and residual tests.

1) Model fitting tests

Model fitting refers to how well a prediction model generalizes the data relative to actual data. A well-fitted model balances accuracy and generalization (subject to residual validation). In contrast, an overfitted model fits previously seen data closely but has trouble generalizing to previously unseen data. This behavior is undesirable and can be easily detected using residual validation techniques. However, residual validation was not performed in the papers that we reviewed.

The model fit category tests are the MSE, One Step Ahead (OSA) and R^2 analysis. OSA measures the ability of a model to predict the future Bitcoin price in the form of Eq. (6):

$$y(t) = g(z(t)) \tag{6}$$

where $g(*)$ is the estimated non-linear model, and $z(t)$ are the model regressors/parameters. In the case of MLP-NARX, $z(t)$ are the weights in the network. R^2 is a measure to indicate the goodness-of-fit of a model. Other works [28] have extended its usage to measure non-linear model fit, originally intended for linear regression problems. The R^2 the measure is calculated as Eq. (7).

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{7}$$

where y_i and $\hat{y}_i$ are actual and estimated observations at interval i, respectively. n is the number of observations and $\bar{y}$ is the mean value of y.

2) Residuals tests

The residuals test tests the randomness of the residuals to ensure it collects all the dynamic datasets. The residual tests are residual plot, histogram analysis, error histogram, and auto- and cross-correlation tests. Ideally, the residuals should appear as white noise – meaning that the model has captured all the dynamics in the system. Residual plots can be used to examine the difference between actual and predicted data, and they are useful for detecting patterns in the residuals. As mentioned previously, these patterns indicate unmodeled dynamics in the data. The model should be improved iteratively until they address these absent dynamics.

A histogram is a graphical method to present a distribution summary of a univariate data set [29]. It is drawn by segmenting the data into equal-sized bins (classes), then plotting the frequencies of data appearing in each bin. The horizontal axis of the histogram plot shows the bins, while the vertical axis depicts the data point frequencies. The histogram analysis is used to view the distribution of the residual. White noise residuals should appear normally distributed with a Gaussian bell-shaped curve.

Correlation tests measure the correlation between two time-series sequences at different points in time. They are useful indicators of dependencies and correlatedness between two sequences. Correlation tests are done by shifting the signals at different lags and measuring the correlation coefficients (degree of correlation). For the purpose of this paper, two important correlation tests are the correlation between the residuals and themselves (autocorrelation of residuals) and the cross-correlation between the residuals and the output. The tests are mathematically described in Eq. (8) and Eq. (9), respectively.

$$\theta_{\varepsilon\varepsilon}(\tau) = E[\varepsilon(t-\tau)\cdot\varepsilon(\tau)] = \delta(\tau) \tag{8}$$

$$\theta_{y\varepsilon}(\tau) = E[y(t-\tau)\cdot\varepsilon(\tau)] = 0 \;\forall\; \tau \tag{9}$$

where $E[\cdot]$ is the mathematical expectation of the correlation function, τ is the lag space, and is Kronecker delta $\delta(\tau)$ defined as Eq. (10):

$$\delta(\tau) = \begin{cases} 0, & \tau \neq 0 \\ 1, & \tau = 0 \end{cases} \tag{10}$$

III. Result And Discussion

A. PSO Training Results

Table 3 shows the PSO optimization. Three MT random seeds were tested to give different starting search points for PSO particles. The results suggest different final solutions for each seed, generally settling at approximately 0.002 MSE. The MSE values for all three solutions appear to approach 0, indicating that the MLP-NARX model optimized the model to reduce the residuals sufficiently.

The best-performing results were at random seed 0 (Table 4). The results show that MLP-NARX model relied heavily on the technical indicators as many past lags were required to achieve the desired results. The best result ($r_{PSO} = 0$) needed less past input and outputs, while the second-best and third-best results utilized more past lags to create the output. The best networks were not particularly large as approximately seven to 24 hidden units appear to achieve good results.

TABLE III
PSO OPTIMIZATION RESULTS

Swarm Size	Max Iteration	Random Seed	Fitness (MSE)
50	100	0	0.000209
50	100	10,000	0.000246
50	100	20,000	0.000260
	Average		0.000238

TABLE IV
OPTIMAL VALUES SELECTED BY PSO AT DIFFERENT PSO RANDOM SEEDS

Parameter	Opt. Value ($r_{PSO} = 0$)	Opt. Value ($r_{PSO} = 10,000$)	Opt. Value ($r_{PSO} = 20,000$)
n_u	1	1	21
n_y	1	15	1
h	24	20	7
$r_{MLP-NARX}$	14,274	4,500	4,057
w_{MA}	27	12	6
w_{BB}	15	30	9
w_{RSI}	24	27	30

B. MLP-NARX Training Performance

The best MLP-NARX training performance plot is shown in Fig. 5. Initially, the MSE value was high due to random initial weights. However, as training progressed, the errors gradually decreased as the MLP weights were updated. The training was stopped at epoch 5 due to ES with a final training MSE score of 5.47×10^{-4}. The training was stopped early to avoid the MLP-NARX from overfitting the training set. ES does this by monitoring the performance of the MLP-NARX on the independent validation set. The training was stopped because the validation error increased consecutively six times (Fig. 5). Any training beyond this point would reduce the generalization of the MLP-NARX.

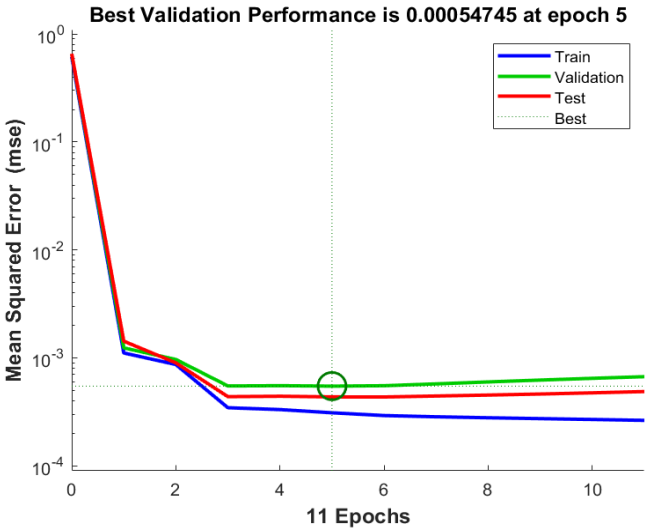

Fig. 5 MLP-NARX training performance

C. One-Step Ahead Prediction

The OSA test was used to assess the model's predictive capacity, as it attempts to forecast one step forward in time based on previous data. Fig. 6 to Fig. 8 show the OSA results for the training, validation, and testing sets, respectively. A large overlap between the actual (blue) and predicted (red) data for training, validation, and testing sets indicates a good agreement between the actual and forecasted data. The model appears to struggle during periods of abrupt price swings as it needs to adjust its predictions slightly from relatively stable past data. This can also be confirmed from the residual plot (Fig. 8), which shows increased residual magnitudes during high price swings.

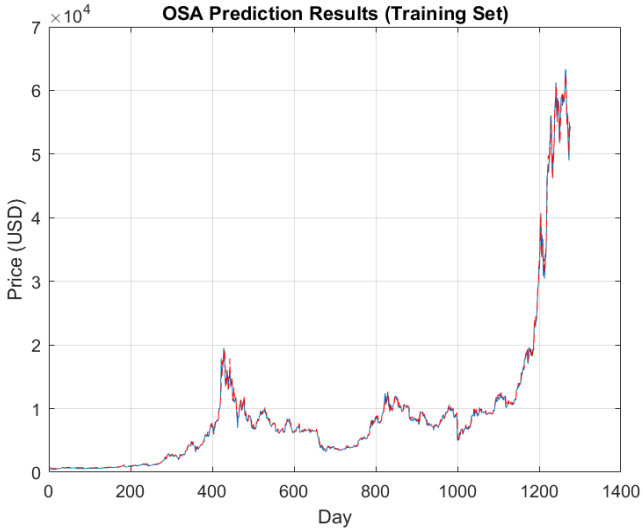

Fig. 6 OSA prediction for a training set

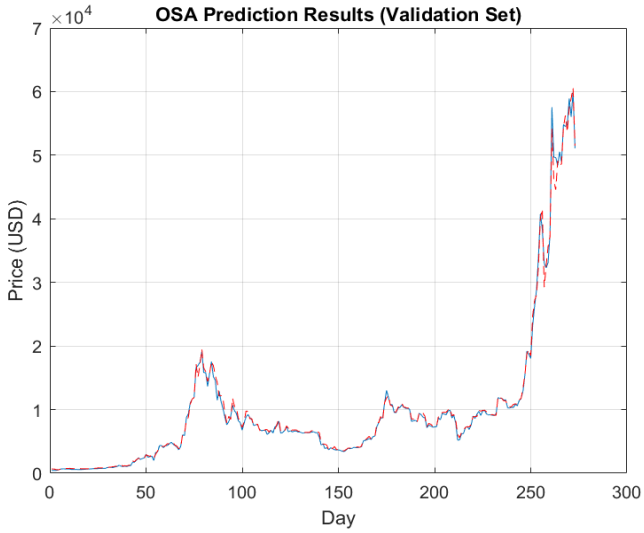

Fig. 7 OSA prediction for the validation set

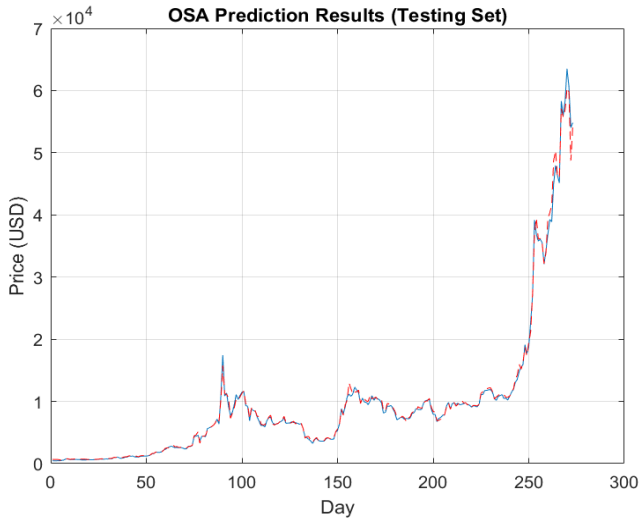

Fig. 8 OSA prediction for the testing set

D. Residual Plot

Fig. 9 shows the residual plot for training, validation, and testing sets. The range of prediction error on all datasets

ranged from approximately -USD 3,000 to USD 4,000. Higher residuals were observed when there were rapid and significant price variations. This is because volatile price fluctuations reduce the model's effectiveness in predicting the output, as the model cannot rely much on historical data to make predictions. Overall, the residuals are not patterned, i.e., systematically high or low. They appear to be centered on zero throughout the range of fitted values, but major bitcoin price swings will dramatically increase the residuals' magnitude.

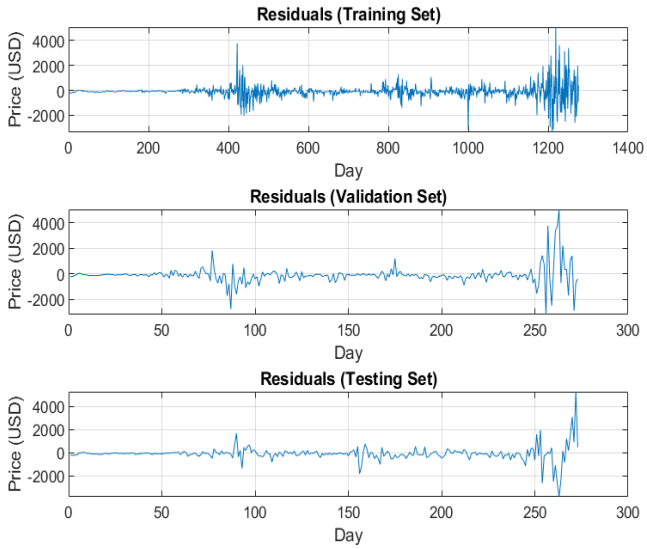

Fig. 9 Residual plot for all datasets

E. *Residuals Analysis*

The randomness of the residuals is important to verify that the model has captured all the dynamics in the dataset. If the model is successful in this task, it should leave behind only white (random and uncorrelated) noise as its residuals. Non-random patterns in the residuals indicate that the model's predictor variables have not sufficiently captured important explanatory information, which manifests in the residuals [30]. Several tests were used to test the whiteness of the residuals – error histogram, autocorrelation, and cross-correlation.

The histogram distribution of the residuals is shown in Fig. 9. White noise distribution should appear as a Gaussian curve with the highest distribution in its center and tapering off at both tails. This observation is evident in Fig. 10.

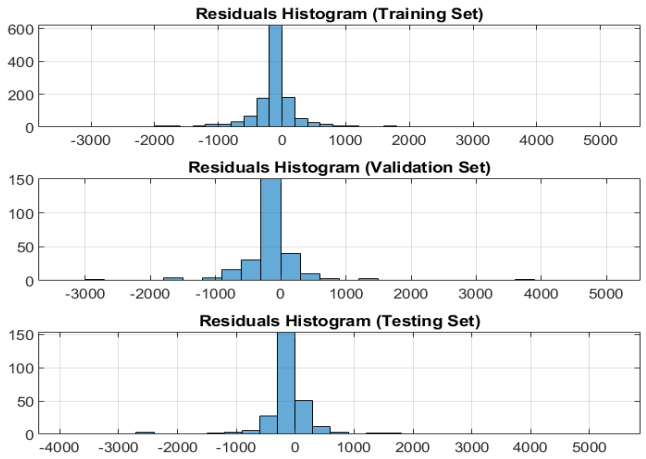

Fig. 10 Residual histogram for all datasets

For the autocorrelation test, the residuals, $\varepsilon(t)$, were compared with themselves. If the residuals are randomly distributed, we expect the correlation coefficient to be one for lag zero and generally within the 95% confidence limits (blue line) for any other lags. Fig. 11 shows the autocorrelation results. Apart from some coefficient violations, all three datasets appear to have passed the abovementioned criteria.

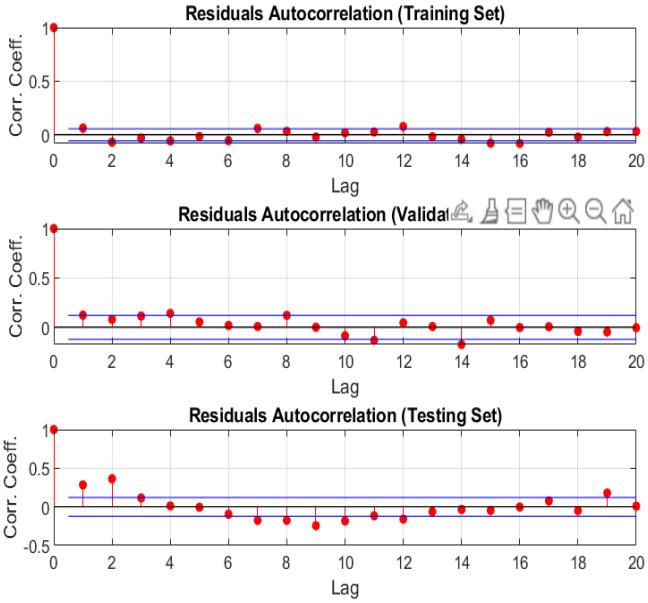

Fig. 11 Residual autocorrelation results

The cross-correlation test compares the output signal, $y(t)$, with the residuals, $\varepsilon(t)$. If the residuals are randomly distributed, we expect the correlation coefficient to be generally within the 95% confidence limits for all lags. Fig. 12 shows the cross-correlation test results. The results indicate that the correlation coefficients were generally within the prescribed limit, albeit with minor violations. These figures concluded that although small information is still present in the residuals, they were largely random. Based on this observation, the residuals were accepted as random noise and subsequently the model was acceptable.

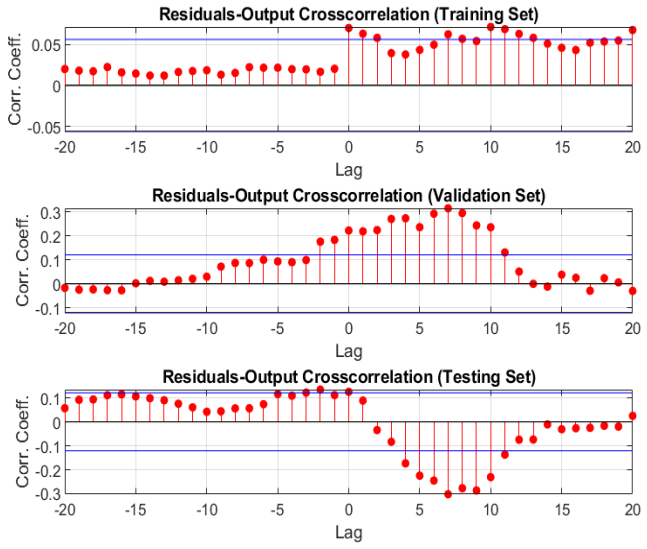

Fig. 12 Residual cross-correlation results

IV. Conclusion

An MLP-NARX prediction model for Bitcoin forecasting is presented in this paper. Objective 2 was addressed by constructing an MLP-NARX model that utilizes historical prices (opening, low, high, closing prices, and trading volume) to help it anticipate the momentum of price fluctuations and evaluate overbought or oversold conditions. Residual analysis (Objective 1) was performed using histograms, autocorrelation, and cross-correlation tests to ensure that the model was unbiased. After running the validation and fitting test, the model had successfully predicted a price one day ahead based on the high number of overlaps in the OSA prediction test. Based on the residuals histogram and correlation test's result, the model has captured most of the dynamics in the dataset by proving the randomness of the residuals. In conclusion, all the tests suggest that the model performed well with minimally correlated residuals. Finally, the PSO algorithm was used to discover optimal parameter values for the model, such as technical indicator window sizes, hidden units, and lag terms to use in the model (Objective 3).

Acknowledgment

The authors thank the Ministry of Higher Education, Malaysia, for equipment support through the Long-term Research Grant Scheme (LRGS) 600-RMC/LRGS 5/3 (001/2020). Financial support from Universiti Teknologi MARA, Malaysia, is gratefully acknowledged.

References

[1] C. Zhou, L. Xing, and Q. Liu, "Dependability Analysis of Bitcoin subject to Eclipse Attacks," *International Journal of Mathematical, Engineering and Management Sciences*, vol. 6, no. 2, pp. 469–479, Apr. 2021, doi: 10.33889/IJMEMS.2021.6.2.028.

[2] Helynda Mulya Arga Retha and Gilberto Daniel Dwi Putra Taslim, "A Forecasting: Bitcoin Price with the ARIMA Method to Help Swing Traders Made Decision," *Internasional Journal of Data Science, Engineering, and Anaylitics*, vol. 2, no. 1, pp. 17–24, May 2022, doi: 10.33005/ijdasea.v2i1.19.

[3] A. Ş. Dokuz, A. Ecemiş, and M. Celik, "Hourly, Daily, and Monthly Analysis of Big Dataset of Bitcoin Blocks," *International Conference on Engineering Technologies*, pp. 100–104, 2019.

[4] P. Urien, "Towards Secure Bitcoin Fast Trading: Designing Secure Elements for Digital Currency Invited Paper," vol. 2, no. August, 2016.

[5] A. van Schetsen, "Impact of graph-based features on Bitcoin prices," Delft University of Technology, Netherlands, 2019.

[6] M. Chen, N. Narwal, and M. Schultz, "Predicting Price Changes in Ethereum," Stanford, CA, 2017.

[7] S. Van Der Avoird, "Bachelor Computer Science Prediction and technical analysis of the Bitcoin," 2020.

[8] Anonymous, "CryptoCurrency Market Capitalizations." https://coinmarketcap.com/.

[9] F. Adjei, "Determinants of *Bitcoin* Expected Returns," *Journal of Finance and Economics*, vol. 7, no. 1, pp. 42–47, 2019, doi: 10.12691/jfe-7-1-5.

[10] N. Stepura, A. Vasyliuk, and I. Makar, "Software for Projecting Bitcoin Course by Artificial Neural Network Methods," in *Proc. 4th Int Conf. Computational Linguistic and Intelligent Systems*, 2020, vol. II, pp. 390–391. doi: 10.15588/1607-3274-2018-4-14.9.

[11] S. A. Alahmari, "Using Machine Learning ARIMA to Predict the Price of Cryptocurrencies," *International Journal of Future Generation Communication and Networking*, vol. 13, no. 1, pp. 745–752, 2020.

[12] Z. Shen, Q. Wan, and D. J. Leatham, "Bitcoin Return Volatility Forecasting: A Comparative Study of GARCH Model and Machine Learning Model," in *2019 Agricultural & Applied Economics Association Annual Meeting*, 2019, pp. 1–20.

[13] B. Barry and M. Crane, "Analysis of cryptocurrency commodities with motifs and LSTM," in *CEUR Workshop Proceedings*, 2019, vol. 2563, pp. 28–39.

[14] A. Misnik, S. Krutalevich, S. Prakapenka, P. Borovykh, and M. Vasiliev, "Comparison of the predictions of convolutional neural networks with image arguments and long short-term memory neural networks with time-series arguments for cryptocurrency markets," *CEUR Workshop Proceedings*, vol. 2475, no. September, pp. 214–222, 2019.

[15] M. McCoy and S. Rahimi, "Towards a Twitter-based Prediction Tool for Digital Currency," in *International Conference of Artificial Intelligence 2019*, 2019, pp. 305–311.

[16] Y. Yang, "Whose opinions prevail on Bitcoin pricing?," University of Helsinki, 2020.

[17] A. A. Maxim, "Prediction of Cryptocurrency Price Movements from Order Book Data Using LSTM Neural Networks," University College London, UK, 2019.

[18] B. Bashir and F. Aslam, "Comparative Analysis of Traditional and Soft Computing for Trading Signals Prediction," *International Transaction Journal of Engineering Management & Applied Sciences & Technologies*, vol. 11, no. 4, pp. 1–16, 2020, doi: 10.14456/ITJEMAST.2020.70.

[19] N. I. Indera, I. M. Yassin, A. Zabidi, and Z. I. Rizman, "Non-linear Autoregressive with Exogeneous input (NARX) bitcoin price prediction model using PSO-optimized parameters and moving average technical indicators," *Journal of Fundamental and Applied Sciences*, vol. 9, no. 3S, p. 791, 2018, doi: 10.4314/jfas.v9i3s.61.

[20] A. Chaudhary, A. Agrawal, and V. Kumar, "Bitcoin Price Prediction using Machine Learning," *International Journal of Multidisciplinary Educational Research*, vol. 9, no. 5(5), pp. 151–156, 2020.

[21] Anonymous, "Daily Bitcoin Prices," 2021. http://www.coindesk.com (accessed Apr. 03, 2021).

[22] R. J. Hyndman, "Moving Averages," in *International Encyclopedia of Statistical Science*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 866–869. doi: 10.1007/978-3-642-04898-2_380.

[23] C. Lento, N. Gradojevic, and C. S. Wright, "Investment information content in Bollinger Bands?," *Applied Financial Economics Letters*, vol. 3, no. 4, pp. 263–267, Jul. 2007, doi: 10.1080/17446540701206576.

[24] A. Zabidi, N. M. Tahir, I. M. Yassin, and Z. I. Rizman, "The performance of Binary Artificial Bee Colony (BABC) in structure selection of polynomial NARX and NARMAX Models," *International Journal on Advanced Science, Engineering and Information Technology*, vol. 7, no. 2, 2017, doi: 10.18517/ijaseit.7.2.1387.

[25] B. A. Amisigo, N. van de Giesen, C. Rogers, W. E. I. Andah, and J. Friesen, "Monthly streamflow prediction in the Volta Basin of West Africa: A SISO NARMAX polynomial modelling," *Physics and Chemistry of the Earth, Parts A/B/C*, vol. 33, no. 1–2, pp. 141–150, Jan. 2008, doi: 10.1016/j.pce.2007.04.019.

[26] D. Nguyen and B. Widrow, "Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights," in *1990 IJCNN International Joint Conference on Neural Networks*, 1990, pp. 21–26 vol.3. doi: 10.1109/IJCNN.1990.137819.

[27] F. Wang, H. Zhang, and A. Zhou, "A particle swarm optimization algorithm for mixed-variable optimization problems," *Swarm and Evolutionary Computation*, vol. 60, p. 100808, Feb. 2021, doi: 10.1016/j.swevo.2020.100808.

[28] B. A. Amisigo, N. van de Giesen, C. Rogers, W. E. I. Andah, and J. Friesen, "Monthly streamflow prediction in the Volta Basin of West Africa: A SISO NARMAX polynomial modelling," *Physics and Chemistry of the Earth, Parts A/B/C*, vol. 33, no. 1–2, pp. 141–150, Jan. 2008, doi: 10.1016/j.pce.2007.04.019.

[29] N. Heckert and J. Filliben, "Chapter 1: Exploratory Data Analysis," in *NIST/SEMATECH e-Handbook of Statistical Methods*, 2003.

[30] I. M. Yassin, "Nonlinear Auto-Regressive Model Structure Selection using Binary Particle Swarm Optimization Algorithm," Universiti Teknologi Mara, 2014.