# A Data Pre-Processing and Machine Learning Approach for Forecasting Crypto Currency Prices

Vicky Soni[1], Prof. C.K.Tiwari[2], Mr. Devkinandan Nagar[3]
M.Tech Scholar, Patel College of Science and Technology, Indore[1]
Associate Professor, Patel College of Science and Technology, Indore[2]
Assistant Professor, Patel College of Science and Technology, Indore[3]

***Abstract*—Machine Learning and Deep Learning Models are being extensively used at the backend of forecasting crypto prices. Many factors, including changes in regulation, public opinion on social media, investor actions, and overall global economic trends, contribute to the high degree of volatility in the cryptocurrency market. Predicting the future value of cryptocurrencies with any degree of accuracy has emerged as a top concern for academics, traders, and investors due to the inherent uncertainty of the market. Machine learning (ML) provides excellent methods for evaluating large volumes of real-time and historical data to forecast price changes in the future. ML models range from simple statistical methods to intricate deep learning architectures. The proposed work employs an optimized second order regularization based back propagation algorithm along with the data pre-processing using the discrete wavelet transform (DWT) for crypto price prediction. It has been shown that the proposed system attains lesser mean square percentage error compared to previously existing technique.**

***Keywords—Machine Learning, Neural Networks, Regularization, Time Series Analysis, Mean Absolute Percentage Error (MAPE).***

## I. INTRODUCTION

Machine learning models rely on extensive and diverse datasets to make reliable cryptocurrency predictions [1]. Sources of data typically include historical price data, volume, open/high/low/close (OHLC) data, order book information, social media sentiment, and macroeconomic indicators. Data preprocessing is essential, given the noisy and unstructured nature of financial data. This involves data cleaning, handling missing values, and transforming data to remove seasonality and reduce volatility. Feature engineering is another critical step, where new variables are created, such as moving averages or relative strength index (RSI), to help models better understand patterns in cryptocurrency price data [2].

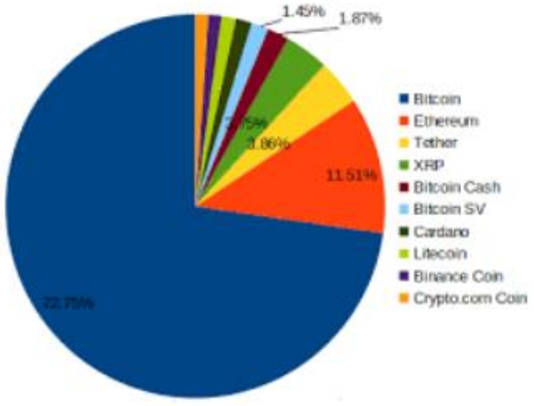

**Fig.1 Percentage Share of Crypto Currencies (Source: [https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9200988](https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9200988))**

raditional time series models, such as Autoregressive Integrated Moving Average (ARIMA), have been used for cryptocurrency forecasting due to their effectiveness in capturing trends and seasonality in historical data. However, ARIMA's linear approach may limit its accuracy in cryptocurrency markets, where nonlinear relationships are common [3]. Long Short-Term Memory (LSTM) networks, a type of recurrent neural network (RNN), are particularly well-suited for sequential data, as they can capture long-term dependencies and nonlinear patterns in time series data. LSTMs have been shown to outperform traditional models in crypto forecasting due to their ability to retain information across time steps, providing better insight into market fluctuations.

Mathematically, the time series crypto price can be modelled as [4]:

$$\boldsymbol{P} = \boldsymbol{f}(\boldsymbol{t}, \boldsymbol{v}) \qquad \textbf{(1)}$$

Here,
P represents crypto price
f represents a function of
t is the time variable
v are other influencing global variables

Despite the potential of machine learning in cryptocurrency forecasting, several challenges persist. The highly volatile and speculative nature of the crypto market makes it challenging to create stable models, as traditional economic indicators may not hold in a decentralized market [5]. Furthermore, the scarcity of long-term data, especially for newer cryptocurrencies, can limit the accuracy of historical-based models. Overfitting is another concern, as cryptocurrency data often exhibits random noise; models trained on such data might perform well on historical data but poorly on future predictions. Finally, the rapid pace of regulatory changes and technological advancements in the cryptocurrency sector means that models must be regularly updated and validated to remain relevant [6].

As cryptocurrency markets continue to grow, advances in machine learning and artificial intelligence will likely lead to even more sophisticated forecasting techniques. Models incorporating real-time data from decentralized exchanges and blockchain analytics may improve forecasting accuracy. Transfer learning, where models trained on similar financial markets are fine-tuned for cryptocurrency data, could help mitigate the data scarcity issue for newer coins. Additionally, federated learning—where models learn from decentralized data without centralized storage—could enhance privacy and security in cryptocurrency forecasting, allowing individual investors and institutions to train models collaboratively while preserving data privacy. [7].

## II. MACHINE LEARNING MODELS

As crypto currency price prediction is a time series analysis problem, hence it is necessary to evaluate the common time series models prior to developing own model. Here the most common time series models which can be used for crypto price forecasting [8].

Time series analysis plays a crucial role in fields ranging from finance and economics to healthcare and environmental science. In time series data, observations are sequentially recorded at specific time intervals, making temporal dependencies a defining feature. Unlike standard supervised learning tasks, time series data often requires models that can account for trends, seasonality, and autocorrelation. Various machine learning models have proven effective for time series analysis, each with unique strengths in capturing these temporal patterns and delivering accurate predictions [9].

Autoregressive Integrated Moving Average (ARIMA)
ARIMA is one of the most widely used statistical models for time series forecasting, especially effective in capturing linear relationships within data. ARIMA models incorporate autoregression (AR), differencing to handle non-stationarity (I for Integrated), and moving averages (MA) to account for residuals. By adjusting its three components (p, d, and q), ARIMA can model many patterns commonly found in time series data, such as trends and seasonality. However, ARIMA is limited to linear relationships and typically requires careful parameter tuning. Despite these limitations, ARIMA remains a valuable tool for time series forecasting when data follows linear trends [10].

Seasonal Decomposition of Time Series (STL) and Seasonal ARIMA (SARIMA)
For data exhibiting seasonality, which is common in many real-world time series, models like SARIMA extend ARIMA by adding a seasonal component. SARIMA adds parameters to ARIMA to account for seasonal cycles at fixed intervals, making it highly effective for time series with repeating seasonal patterns. Seasonal decomposition, or STL (Seasonal and Trend decomposition using Loess), is another approach that separates a time series into its trend, seasonality, and residual components. These decomposition techniques allow for more accurate analysis by addressing the distinct characteristics of each component, providing models with cleaner inputs for prediction tasks [11].

Exponential Smoothing (ETS)
Exponential smoothing techniques are another category of statistical methods for time series forecasting. These models, such as the Holt-Winters Exponential Smoothing, apply exponentially decreasing weights to past observations, meaning more recent data points have a greater influence on predictions. ETS models account for trends and seasonality, making them suitable for data with level shifts and cyclical patterns. Because of their simplicity and effectiveness in certain cases, exponential smoothing methods are often used as baseline models for time series forecasting. While ETS models are straightforward and computationally efficient, they may underperform on highly complex datasets with nonlinear relationships [12].

Long Short-Term Memory (LSTM) Networks
LSTM networks, a type of recurrent neural network (RNN), are widely used for time series forecasting due to their ability to capture long-term dependencies in sequential data. LSTM models feature memory cells that retain information over extended sequences, making them particularly well-suited for time series data with long-term trends. By managing information through gates (input, forget, and output), LSTMs can learn which information to keep or discard, adapting to both short-term and long-term patterns. LSTMs have been successfully applied in fields like stock market prediction and weather forecasting, where complex, nonlinear patterns are common. However, they are computationally intensive and require large datasets to avoid overfitting [13].

Gated Recurrent Units (GRU)
GRUs are a variant of LSTM networks designed to be simpler and computationally more efficient while still handling sequential dependencies effectively. GRUs have fewer gates than LSTMs, making them faster to train and less prone to overfitting on smaller datasets. Despite their simplified structure, GRUs often deliver comparable performance to LSTMs in time series forecasting tasks, especially when the dataset is smaller or requires shorter-term forecasting. GRUs are frequently used in applications where computational resources are limited, or where faster model training is a priority, balancing predictive power with efficiency [14].

XGBoost and Random Forest for Time Series
While not originally designed for time series data, ensemble methods like XGBoost and Random Forest have been adapted for time series forecasting. By transforming time series data into a supervised learning format, these models can capture complex relationships and handle nonlinearity. XGBoost, in particular, is powerful for capturing intricate patterns by constructing an ensemble of decision trees in a gradient boosting framework. Random Forest, on the other hand, averages predictions from multiple decision trees, reducing overfitting and capturing trends in noisy data. These models are advantageous in situations where temporal dependencies are weaker or when additional explanatory variables are present, as they can easily integrate external features [15].

Convolutional Neural Networks (CNNs) for Time Series Analysis
Although CNNs are typically associated with image data, they have been successfully applied to time series forecasting, particularly through 1D convolutions. CNNs are capable of extracting local patterns, which can be beneficial for time series data with short-term dependencies. CNNs are often used in combination with LSTMs or GRUs to form hybrid models, where CNN layers extract short-term patterns, and recurrent layers capture longer-term dependencies. These hybrid models have shown promise in fields like energy demand forecasting and predictive maintenance, where capturing both local and long-term patterns is essential for accurate forecasts.
Hybrid Models: Combining Strengths for Enhanced Performance [16]
Hybrid models, which integrate multiple machine learning techniques, have gained popularity for time series forecasting due to their ability to capture a wider range of patterns. For example, an LSTM network might be combined with an attention mechanism to selectively focus on important time steps, or a CNN layer could be added to capture local dependencies before passing data to a recurrent layer. These hybrid models leverage the strengths of each component model, providing more flexible and accurate

predictions. Hybrid models are particularly useful in complex time series data with both short-term and long-term dependencies, offering robust performance across a range of applications [17].

## III. PROPOSED ALGORITHM

The proposed algorithm majorly comprises of two main approaches:

***Data Filtration:***

The discrete wavelet transform or DWT has been used for data filtration. The wavelet transform is an effective tool for removal of local disturbances. Crypto prices show extremely random behavior and local disturbances. Hence conventional Fourier methods do not render good results for highly fluctuating data sets. Mathematically, the wavelet transform can be given as [18]

$$Z(S,P) = \int_{-\infty}^{\infty} z(t)\,((S,P,t))dt \qquad (2)$$

Here,
S denotes the scaling operation
P denotes the shifting operation
t denotes the time variable
Z is the image in transform domain
z is the image in the spatial domain

The major advantage of the wavelet transform is the fact that it is capable of handling fluctuating natured data and also local disturbances. The DWT can be defined as [19]:

$$W\Phi\,(Jo, k) = \frac{1}{\sqrt{M}}\sum_{n} S(n).\Phi(n)_{jo,k} \qquad (3)$$

***Optimized Back Propagation Model:***

Back propagation is one of the most effective ways to implement the deep neural networks with the following conditions:

1) Time series behavior of the data
2) Multi-variate data sets
3) Highly uncorrelated nature of input vectors

The essence of the back propagation based approach is the fact that the errors of each iteration is fed as the input to the next iteration. [20]. The error feedback mechanism generally is well suited to time series problems in which the dependent variable is primarily a function of time along with associated variables. Mathematically [21],

$$Y = f(t, V_1 \ldots . V_n) \qquad (4)$$

Here,
Y is the dependent variable
f stands for a function of
t is the time metric
V are the associated variables
n is the number of variables
The back propagation based approach can be illustrated graphically in figure 2.

In case of back propagation, the weights of a subsequent iteration doesn't only depend on the conditions of that iteration but also on the weights and errors of the previous iteration mathematically given by [22]:

$$W_{k+1} = f(W_k, e_k, V) \qquad (5)$$

Here,
$W_{k+1}$ are the weights of a subsequent iteration
$W_k$ are the weights of the present iteration
$e_k$ is the present iteration error
V is the set of associated variables
In general, back propagation is able to minimize errors faster than feed forward networks, however at the cost of computational complexity at times. However, the trade off between the computational complexity and the performance can be clearly justified for large, complex and uncorrelated datasets for data sets. The optimized regularization based back propagation model can be trained using the following training rule [23]:

$$w_{k+1} = w_k - \left[J_k J_k^T + \mu' I\right]^{-1} J_k^T e_k \qquad (6)$$

Here,

K is the iteration number

$w_{k+1}$ is weight of next iteration,

$w_k$ is weight of present iteration

$J_k$ is the Jacobian Matrix and is given by the terms $J_k = \frac{\partial^2 e}{\partial w^2}$ i.e. the second order rate of change of errors with respect to weights

$J_k^T$ is Transpose of Jacobian Matrix

$e_k$ is error of Present Iteration

$\mu'$ is step size i.e. amount by which weight changes in each iteration

$I$ is an identity matrix, with all diagonal elements equal to 1 and other elements 0.

The regularization term $\rho$ updates the weight through step sizes of $\mu' = \rho\mu$ so as to minimize the cost function without the chances of overfitting or saddle points [24]. The data is divided in the ration of 70:30 for training and testing data set bifurcation.
The final performance metrics computed for system evaluation are:
1) Mean Absolute Percentage Error (MAPE)

$$MAPE = \frac{100}{M}\sum_{t=1}^{N}\frac{E-E_t|}{E_t} \qquad (7)$$

Here $E_t$ and $E_t^{\sim}$ stand for the predicted and actual values respectively.
The number of predicted samples is indicated by M.

2) Regression
The extent of similarity between two variables is given by the regression.

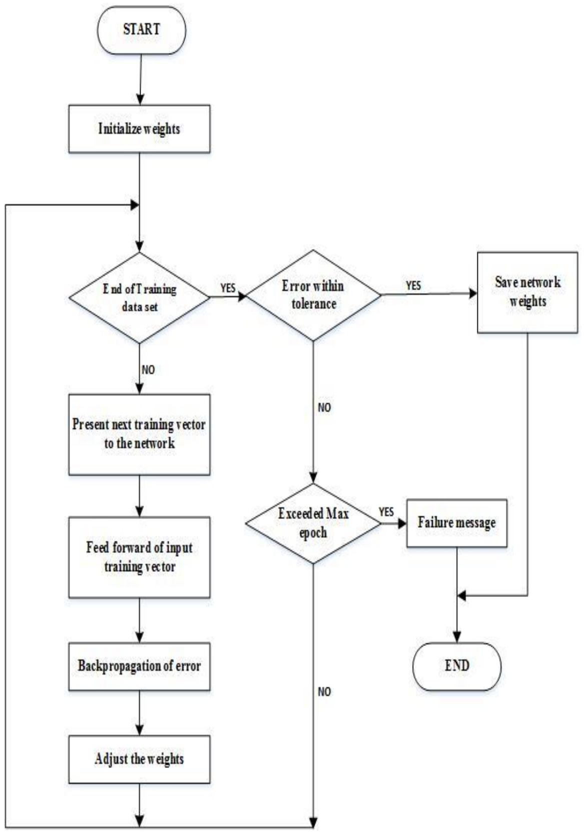

**Fig.2 Flowchart of Back Propagation**

The proposed algorithm is presented next:

```
*Start*
*{*
*Step.1* Split data into the ratio of 70:30 for training : testing.
*Step.2* Apply DWT for filtration
*Step.3* Initialize weight matrix randomly.
*Step.4* To train the network, employ the following training rule:
        $w_{k+1} = w_k - \left[J_k J_k^T + \mu' I\right]^{-1} J_k^T e_k$
*Step.5* If (cost function stabilizes)
        Truncate training
        Else if (max. iterations are over)
        Truncate Training
        Else
        Feedback errors as inputs to subsequent iteration.
*Step.6* if (error is stable through validation checks i.e. consecutive iterations)
        Stop training
        else if (maximum iterations are over even without error stabilization)
        Stop Training
        else
        {
        Feed next training vector
        Back propagation of error
        }
Step.7: Simulate model to forecast samples.
Step.8 Compute performance metrics.
}
```

## III. RESULTS

The simulations have been performed on MATLAB. The data source is obtained from Yahoo Finance: https://finance.yahoo.com/quote/BTC-USD/

The libraries used are the statistical and deep learning libraries

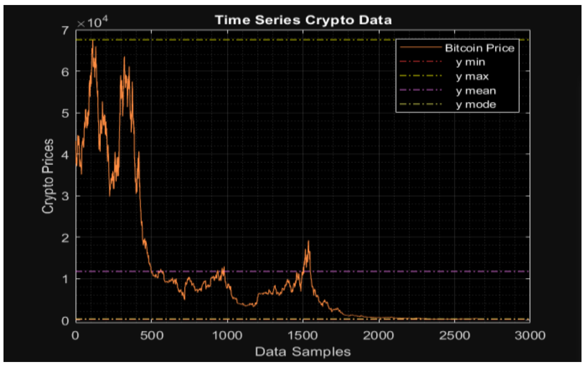

**Fig.3 Original Bitcoin Price**

The figure above depicts variation in Bitcoin price.

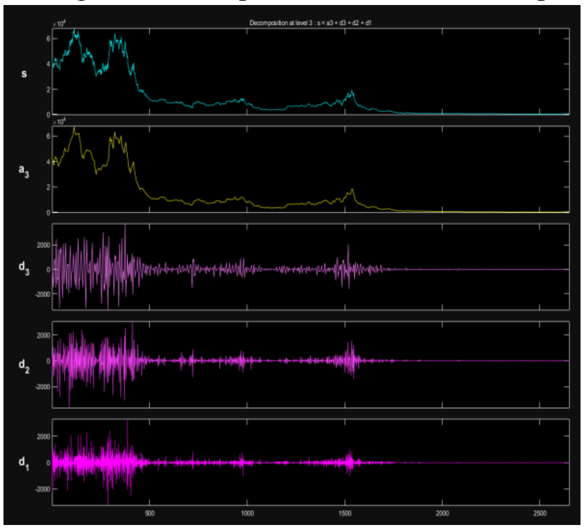

**Fig.4 Haar Decomposition at Level 3**

The figure above depicts the Haar decomposition in terms of approximate and detailed co-efficients for the data.

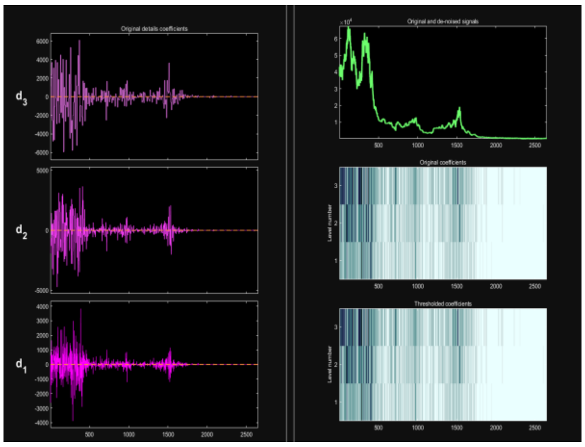

**Fig.5 Denoising data using DWT**

The figure above depicts the denoising process using Haar

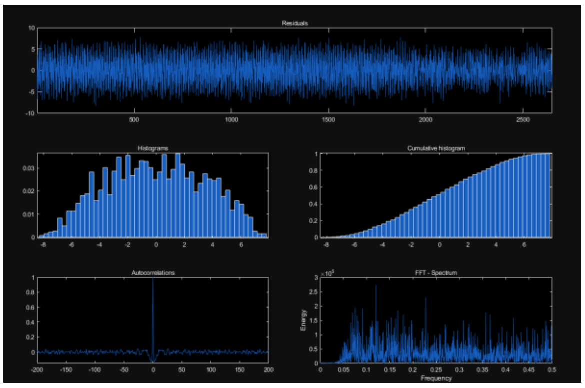

**Fig.6 Multi-Resolution Analysis**

The figure above depicts the multi resolution analysis of noise baseline.

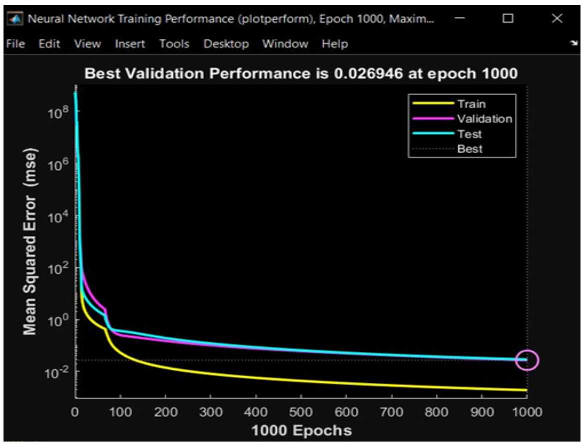

**Fig.7 MSE to Convergence**

The figure above depicts the MSE to convergence for the model

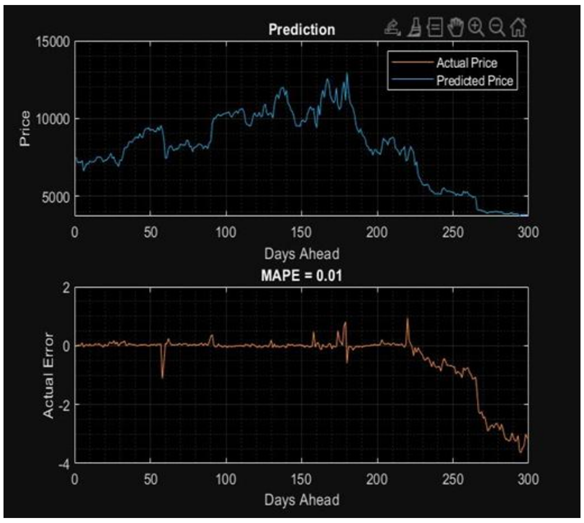

**Fig.8 Predicted and Actual Crypto Behavior**

The figure above depicts the predicted and actual crypto behavior.
The summary of results is presented next:

**Table 1. Summary of Results**

| **S.No.** | **Parameter** | **Value** |
|---|---|---|
| 1. | Crypto Dataset | Bitcoin |
| 2. | Data source | Yahoo Finance |
| 3. | Data Pre-Processing | DWT |
| 4. | ML Model | Deep Neural Network |
| 5. | Algorithm | Bayesian Optimization |
| 6. | Iterations to Convergence | 1000 |
| 7. | MSE to Convergence | 0.026 |
| 8. | Accuracy (Han et al. [11]) | MAPE of 2.66 |
| 9. | Patel et al. [12] | MAPE of 2.7. |
| 10. | Rafi et al. [13] | MAPE of 0.66 |
| 11. | Kim et al. [14] | MAPE of 1.3251 |
| 12. | Shahbazi et al. [15] | MAPE of 3.16 |
| 13 | Mudassir et al. [16] | MAPE of 1.44 |
| 14. | **Proposed Approach** | **MAPE of 0.01** |

It can be observed that the model attains convergence at 1000 iterations with an MSE value of 0.026 and MAPE of just 0.01 outperforming existing work in the domain.

## CONCLUSION

**Crypto price as a time series forecasting model presents unique challenges due to its sequential nature and the need to account for temporal dependencies. Machine learning offers a suite of models suited to different aspects of time series data, from traditional statistical models like Each model brings distinct advantages, whether in simplicity, computational efficiency, or the ability to capture nonlinear patterns. Choosing the right model depends on the characteristics of the dataset, the desired forecast horizon, and the specific application. This research work presents an optimized regularization based approach for crypto price forecasting. The model attains a forecasting MAPE of 0.02% only, much lesser than existing approaches in the domain.**

## REFERENCES

1. N. P. Patel et al., "Fusion in Cryptocurrency Price Prediction: A Decade Survey on Recent Advancements, Architecture, and Potential Future Directions," in IEEE Access, vol. 10, pp. 34511-34538, 2022.
2. S. Karasu, A. Altan, Z. Saraç and R. Hacioğlu, "Prediction of Bitcoin prices with machine learning methods using time series data," 2018 26th Signal Processing and Communications Applications Conference (SIU), Izmir, Turkey, 2018, pp. 1-4.
3. Y. Indulkar, "Time Series Analysis of Cryptocurrencies Using Deep Learning & Fbprophet," 2021 International Conference on Emerging Smart Computing and Informatics (ESCI), Pune, India, 2021, pp. 306-311.

4. M. Saad, J. Choi, D. Nyang, J. Kim and A. Mohaisen, "Toward Characterizing Blockchain-Based Cryptocurrencies for Highly Accurate Predictions," in IEEE Systems Journal, 2020, vol. 14, no. 1, pp. 321-332.
5. P. Jay, V. Kalariya, P. Parmar, S. Tanwar, N. Kumar and M. Alazab, "Stochastic Neural Networks for Cryptocurrency Price Prediction," in IEEE Access, 2020, vol. 8, pp. 82804-82818.
6. P. Jay, V. Kalariya, P. Parmar, S. Tanwar, N. Kumar and M. Alazab, "Stochastic Neural Networks for Cryptocurrency Price Prediction," in IEEE Access, vol. 8, pp. 82804-82818, 2020.
7. S. Otabek and J. Choi, "From Prediction to Profit: A Comprehensive Review of Cryptocurrency Trading Strategies and Price Forecasting Techniques," in IEEE Access, vol. 12, pp. 87039-87064, 2024
8. H. Jang and J. Lee, "An Empirical Study on Modeling and Prediction of Bitcoin Prices With Bayesian Neural Networks Based on Blockchain Information," in IEEE Access, vol. 6, pp. 5427-5437, 2018.
9. B. H. A. Khattak et al., "A Systematic Survey of AI Models in Financial Market Forecasting for Profitability Analysis," in IEEE Access, vol. 11, pp. 125359-125380, 2023.
10. H. Shamshad, F. Ullah, A. Ullah, V. R. Kebande, S. Ullah and A. Al-Dhaqm, "Forecasting and Trading of the Stable Cryptocurrencies With Machine Learning and Deep Learning Algorithms for Market Conditions," in IEEE Access, vol. 11, pp. 122205-122220, 2023.
11. P Han, H Chen, A Rasool, Q Jiang, M Yang, “MFB: A Generalized Multimodal Fusion Approach for Bitcoin Price Prediction Using Time-Lagged Sentiment and Indicator Features”, Expert Systems with Applications, Elsevier 2025, vol.261, 125515.
12. R Patel, J Chauhan, NK Tiwari, V Upaddhyay, A Bajpai, “A deep learning framework for hourly bitcoin price prediction using bi-LSTM and sentiment analysis of Twitter data”, SN Computer Science, Springer 2024, vol.57, no.767.
13. M. Rafi, Q. A. K. Mirza, M. I. Sohail, M. Aliasghar, A. Aziz and S. Hameed, "Enhancing Cryptocurrency Price Forecasting Accuracy: A Feature Selection and Weighting Approach With Bi-Directional LSTM and Trend-Preserving Model Bias Correction," in IEEE Access, vol. 11, pp. 65700-65710, 2023
14. G. Kim, D. -H. Shin, J. G. Choi and S. Lim, "A Deep Learning-Based Cryptocurrency Price Prediction Model That Uses On-Chain Data," in IEEE Access, vol. 10, pp. 56232-56248, 2022
15. Z. Shahbazi and Y. -C. Byun, "Improving the Cryptocurrency Price Prediction Performance Based on Reinforcement Learning," in IEEE Access, vol. 9, pp. 162651-162659, 2021.
16. M Mudassir, S Bennbaia, D Unal, M Hammoudeh, “Time-series forecasting of Bitcoin prices using high-dimensional features: a machine learning approach”, Neural computing and applications, Springer 2020.
17. J.-Z. Huang, W. Huang, and J. Ni, ‘‘Predicting bitcoin returns using highdimensional technical indicators,’’ J. Finance Data Sci., vol. 5, no. 3, pp. 140–155, Sep. 2019.
18. R. Adcock and N. Gradojevic, ‘‘Non-fundamental, non-parametric bitcoin forecasting,’’ Phys. A, Stat. Mech. Appl., vol. 531, Oct. 2019, Art. no. 121727.
19. D. Philippas, H. Rjiba, K. Guesmi, and S. Goutte, ‘‘Media attention and bitcoin prices,’’ Finance Res. Lett., vol. 30, pp. 37–43, Sep. 2019.
20. J. Abraham, D. Higdon, J. Nelson, and J. Ibarra, ‘‘Cryptocurrency price prediction using tweet volumes and sentiment analysis,’’ SMU Data Sci. Rev., vol. 1, no. 3, p. 1, 2018.
21. D. Shen, A. Urquhart, and P. Wang, ‘‘Does Twitter predict bitcoin?’’ Econ. Lett., vol. 174, pp. 118–122, Jan. 2019.
22. AC Wilson, R Roelofs, M Stern, “The marginal value of adaptive gradient methods in machine learning”, Proceedings in Advances in Neural Information Processing Systems 30 (NIPS 2017).
23. L. V. Jospin, H. Laga, F. Boussaid, W. Buntine and M. Bennamoun, "Hands-On Bayesian Neural Networks—A Tutorial for Deep Learning Users," in IEEE Computational Intelligence Magazine, vol. 17, no. 2, pp. 29-48, May 2022

24. M Devadas, V Hiremani, "Integrating dropout and kullback-leibler regularization in Bayesian neural networks for improved uncertainty estimation in regression", MethodsX, Elsevier 2024, vol.12, 102659.