University of Piraeus

School of Information and Communication Technologies

Department of Informatics

Thesis

Τίτλος Title	Πρόβλεψη κρυπτονομίσματος με LSTM νευρωνικά δίκτυα Cryptocurrency price prediction using LSTM neural networks
Student fullname	Ioannis Dosiadis
Father name	Athanasios
Student ID	Π20059
Supervisors	Dionysios Sotiropoulos, Georgios Tsixrintzis

Month/Year: September 2024

Copyright ©

Abstract

Cryptocurrencies have been growing in interest, among all sorts of people and mostly investors. Above all cryptocurrencies, Bitcoin shows the highest amount of price volatility, driven by numerous political and economic reasons that make the trajectory hard to predict. Therefore, this paper tries to find out the efficiency of an LSTM neural network pre-processed with some data in forecasting the next day closing prices of Bitcoin. From our results, this LSTM model was able to efficiently predict the prices of Bitcoin with a mean absolute percentage error of 0.07. Moreover, standardization and normalization were applied, elevating its predictive power for the model enormously.

Keywords: forecasting, cryptocurrency, LSTM, neural networks, data preprocessing

ΠΕΡΙΛΗΨΗ

Τα κρυπτονομίσματα έχουν αυξήσει το ενδιαφέρον, μεταξύ όλων των ανθρώπων και κυρίως των επενδυτών. Περισσότερο από όλα τα υπόλοιπα κρυπτονομίσματα, το Bitcoin παρουσιάζει το υψηλότερο ποσοστό αστάθειας των τιμών, λόγω πολλών πολιτικών και οικονομικών λόγων που καθιστούν δύσκολη την πρόβλεψη της τροχιάς. Ως εκ τούτου, αυτή η έρευνα προσπαθεί να ανακαλύψει την αποτελεσματικότητα ενός νευρωνικού δικτύου LSTM που έχει υποστεί προεπεξεργασία με ορισμένα δεδομένα στην πρόβλεψη των τιμών κλεισίματος της επόμενης ημέρας του Bitcoin. Από τα αποτελέσματά μας, το μοντέλο LSTM ήταν σε θέση να προβλέψει αποτελεσματικά τις τιμές του Bitcoin με μέσο απόλυτο ποσοστό σφάλματος (MAPE) 0,07. Επιπλέον, εφαρμόστηκαν τυποποίηση και κανονικοποίηση, αυξάνοντας την προγνωστική του ισχύ για το μοντέλο πάρα πολύ.

Λέξεις-κλειδιά: πρόβλεψη, κρυπτονομίσματα, LSTM, νευρωνικά δίκτυα, προεπεξεργασία δεδομένων

Contents

1. Introduction ... 1

2. Literature review ... 3

2.1 Traditional time series methods ... 3

2.2 Neural Networks Methods ... 4

2.3 Single deep learning methods ... 5

2.4 Ensemble and Machine learning methods ... 7

2.5 Hybrid machine and deep learning Methods ... 8

2.6 Literature review summary ... 9

3. Methodology ... 10

3.1 Methodology Strategy ... 10

3.2 Data Collection ... 13

3.3 Features Description ... 13

3.4 Feature Selection ... 14

3.5 Train/Test Data ... 14

3.6 Data Pre-processing ... 15

3.6.1. Feature Scaling ... 15

3.6.2. Data Representation of Neural Networks ... 16

3.7 Modelling ... 17

3.7.1. LSTM network ... 17

3.7.2. Final models Implementation ... 23

3.8 Evaluation of deep learning models ... 25

3.8.1. Mean Absolute Percentage Error ... 25

4. Data and Results ... 26

4.1 Descriptive statistics ... 26

4.2 Feature Selection ... 28

4.3 Hyper-parameters tuning ... 29

4.3.1. Number of hidden layers ... 29

4.3.2. Number of units in the LSTM hidden layer 30
4.3.3. Learning rate....31
4.4 Dropout regularization 32
4.5 Final LSTM model evaluation 33
5. Discussion....36
5.1 Key Findings 36
5.2 Study Limitations....36
5.3 Limitations from studies 36
5.4 Strengths 37
5.5 Incidental observation....37
5.6 Comparison with deep learning models included in the review 38
5.7 Future research 39
6. Conclusion....40
References 42

1. Introduction

In two decades of Bitcoin existence (Chiu & Keister, 2022), several thousands of digital coins and hundreds of exchanges have appeared. This has made digital currencies subjects of interest to investors, regulators and the public (Giudici et al., 2020). In no time, markets and activities around such currencies grew rapidly to include online trading platforms, crypto-based derivatives trading, and crypto lending platforms. Central banks are also piloting central bank digital currencies in some countries with a view of introducing them into the economy in future. But there are also market concerns about this phenomenon.

Public debates on cryptocurrencies' high volatility and lack of intrinsic worth have been the subject matter for researchers from various fields (Giudici et al., 2020). Concerns that these are bubble-like assets with no underlying value and facilitating tax evasion can prompt governments to take more stringent measures. This speculation complicates forecasting cryptocurrency prices thereby making it a crucial topic for research worldwide.

Bitcoin is in the lead of cryptocurrency, with it's market capitalization of over $1,343,883,308,328. Its value leverages blockchain technology for extensive digital circulation (Coinmarketcap, 2024). Researchers have historically used classical, statistical, and financial methods like autoregressive integrated moving average (ARIMA) and generalized autoregressive conditional heteroscedasticity (GARCH) to predict bitcoin prices (Gradojevic et al., 2021).

The increased computational power and deep learning created with the aid of machine learning algorithms have given the ability to develop new models that predict the price of bitcoin. Algorithms such as artificial neural networks (ANN), convolutional neural networks (CNN), recurrent neural networks (RNN), and long short-term memory (LSTM) networks are suitable for forecasting bitcoin prices.

The goal of this study is to find out if LSTM neural networks are efficient in bitcoin price predictions. We suggest that the bitcoin closing price for tomorrow be predicted by using

LSTM models. To do this, we obtained historical prices for BTC-USD, preprocessed it and trained an LSTM model on it. As well as that, different hyperparameters were also tried such as number of hidden layers, units per layer, batch size, optimizer, learning rate and dropout regularization to enhance the model's predictive power. Lastly, we present the predictions which our model made with the real price of bitcoins so that we can measure its accuracy.

The following is the systematic literature review on cryptocurrency forecasting which is followed by the Chapter 3 where methodologies of predicting the closing prices of Bitcoin with their performance metrics are discussed. Descriptive statistics of the data, findings, and results from experiments are found in Chapter 4 and consequently, in detail, Chapter 5 explains the results and analyses.

2. Literature review

This section gives an overview of the literature review done. Our target was to analyze research that is currently being done on financial forecasting, more so on cryptocurrency. We have gone through several tools and methodologies; among them include statistical and time series analysis, deep learning, recurrent neural networks, ensemble machine learning techniques.

2.1 Traditional time series methods

This section will shortly discuss traditional research in the field of time series forecasting.

Septiarini et al. (2020) proposed in their research a model that combined conventional statistical techniques with artificial intelligence methods in predicting Bitcoin prices. They used, among others, time series models such as Autoregressive Integrated Moving Average and Exponential Smoothing, along with some AI methods like Fuzzy Time Series and Adaptive Neuro-Fuzzy Inference System. Of course, the best performance was obtained with statistical methods, and AI methods were dominated by exponential smoothing, evidenced by the lowest (RMSE) and (MSE).

Tan and Kashef (2019) have compared machine learning and deep learning techniques with ARIMA statistical methods to predict Bitcoin price. For each transaction, five inputs were used: open, high, low, close prices, and transaction volume. In this comparative study, some of the applied methods were Bayesian Regression, Auto Regression, Long Short-Term Memory, and Support Vector Machines. It was observed from the outcomes of this technique that the LSTM algorithm shows better utility than the rest, with SVM and ARIMA pacing afterward in terms of performance.

Munim et al. (2019), using the ARIMA and the Neural Network Autoregression, attempted to predict the likely Bitcoin price on the following day as a univariate model. These models feature tests with re-estimation forecasts and without, two training sets, and two test sets for cross-validation. Their results showed that ARIMA models performed consistently better than the NNAR models when testing forecasts. That the reasons for explaining the superior performance of ARIMA was to be a feedforward type, though the results could have been further enhanced with much backpropagation. Further, forecasts of ARIMA were the same whether model re-estimation is employed.

This survey of classical time series forecast studies suggests methods such as ARIMA will be able to secure low prediction errors, and they might even more often outstrip deeper and more complicated learning models. This could be attributed to the sensitivity of the data and the effective process of data collection and preparation for algorithm operation. Sometimes, even the shallow and simple-sounding traditional methods outperform sophisticated deep learning techniques. Septiarini et al. (2020) add that modern times models of all the modern advancements cannot always guarantee the best forecasting results since in most cases the characteristics of each dataset are unique.

2.2 Neural Networks Methods

This section reviews the methods and algorithms of various research papers that relate to neural network forecasting.

Radityo et al. (2018) conducted a comparison among some Artificial Neural Network techniques to forecast the closing price of Bitcoin for the next day. They considered BPNN, or backpropagation neural network; GANN, or genetic algorithm neural network; GABPNN, or genetic algorithm backpropagation neural network; and NEAT, or neuroevolution of augmenting topologies. The MAPE returned by GABPNN in the study was best at 1.88%. However, the training time for GABPNN was impractically long for application on larger datasets. In contrast, BPNN returned a MAPE of 1.98% but was three times faster. This research had contributed to the field by comparing several methods of ANNs and considering an important factor like training time. It suggested that slightly higher prediction errors could be tolerable, but they may be preferred if they lead to the development of algorithms that are less complex and require less time to run.

In connection with model parameterization, another pertinent factor for performance, Jay et al. (2020) proposed a stochastic neural network model derived from random walk theory for the prediction of cryptocurrency prices. In their case, they used a multi-layer perceptron model in which they introduced randomness at layer-wise level into the activation features of the network to simulate market volatility. It led to the conclusion that, overall, the stochastic models outperform the deterministic ones. In this regard, a study goes on to say that, although their technique worked well enough, there is still some room for enhancement that can be attained with even better tuning of the hyperparameters in a more precise manner.

2.3 Single deep learning methods

This section lays down the methods and algorithms applied in the respective studies chosen within the domain of single-deep-learning forecasting.

Popular deep learning models for analyzing time sequences with varying lengths, seeking to make predictions, include Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs), and Long Short-Term Memory (LSTM) nets.

Deep learning has been one of the most promising techniques in forecasting cryptocurrency prices. For an example, Ferdiansyah et al., 2019, developed an LSTM-based model in predicting Bitcoin prices. The data that they used for training is four-year historical data and then tested on one-year data. Their promising model was able to give a prediction of next-day Bitcoin prices, though the author considered it to be not proper for making investments as the RMSE is high.

Preparation of data and quality is the most important factor in the implementation of deep learning algorithms. Rizwan et al. (2019) developed a multivariate deep learning model series with both LSTM and Gated Recurrent Units. Such features included Bitcoin exchange rates, trading volumes, transaction fees, and average hash rates. The authors also claimed that exogenous variables for cryptocurrency prices could be indicators of the international economy. For example, from the results, only proper parameterization with high-quality data gave correct predictions for the model.

Selecting useful variables from available data will also be key in the performance of deep learning. Lamothe-Fernandez et al. studied a comparison between the methods of prediction in the price of Bitcoin. It became known that new important variables' sets increase the performance of models, giving the model greater stability over time frames of one and two years and best performance for the Dynamic Convolutional Neural Network.

The recent literature on LSTM has dealt with problems of time series, and Lahmiri and Bekiros studied the ability of LSTM nets in detecting chaotic and self-similar patterns exhibited by the top three cryptocurrencies, namely Bitcoin, Ripple, and Digital Cash. They claimed that LSTM networks are suitable for both short and long-term predictions because of the ability to identify hidden patterns in nonlinear and chaotic data.

There has been little research on real-time cryptocurrency price prediction. Zoumpekas et al. (2020) demonstrated a web-based real-time Ethereum price predictor using the LSTM model. Their system predicts in half-hour intervals on 30 minutes' worth of previous data, proving the LSTM and GRU models to be real-time-application-suited.

GRU networks have fewer parameters, thus being less computationally expensive for training than LSTMs, and consequently, they take less training time and, in general, perform better. Phaladisailoed and Numnonda (2018) compared Huber Regression, LSTM, and GRU in modeling Bitcoin price forecasting. Their study showed that among the models tested, GRU had the best accuracy and convergence time but added that inclusion of more explanatory variables would further make it better with more performance.

Public attention and macroeconomic factors also play a core part in the determination of cryptocurrency value. Public attention, cryptocurrency market factors, and macroeconomic indicators were integrated by Liu et al. (2021) using 40 determinants in an SDAE model. Their model works quite better than SVR and BPNN, and this great result is attached to the great variety of features that have been included.

It is identified that sentiment analysis and public opinion are indispensable in changing the future forecasting models related to cryptocurrency. When social media comments are considered in forecasting models, the level of accuracy improves. In agreement, the research showed that even though LSTM is the most accurate, the addition of social sentiment features by participants made other models to be scored high.

According to the literature, in general, methods that involve deep learning, such as LSTM and GRU, are effective approaches toward cryptocurrency price prediction. So it follows how important the data quality and choice of relevant variables is to the performance of models, including those economic and social sentiment variables.

2.4 Ensemble and Machine learning methods

This section considers some of the techniques and algorithms used in many research papers dealing with ensemble and machine learning forecasting.

Ensemble machine learning is only one type of learning that allows systems to make predictions from very varying models to improve general accuracy. Generally, it is a technique combining variable predictions or the different model outputs for coming up with better results than what the model would have performed alone. Derbentsev et al. (2021) conducted a comparative study on the performance of ensemble machine learning algorithms for cryptocurrency price prediction. They used random forests and a stochastic gradient boosting machine to predict the prices for Bitcoin, Ethereum, and Ripple. The results are rather promising: in reality, for the three cryptocurrencies, SGBM and RF achieved MAPEs between 0.92% and 2.61%. In more detail, SGBM worked well for the prediction of Bitcoin and Ripple, while RF did so for that of Ethereum.

Mallqui and Fernandes performed a further performance-based experiment on Artificial Neural Networks and Support Vector Machines in regard to price prediction for Bitcoins. Experimental results have also noticed an algorithm that gives the most accurate, by which MAPE for each and every prediction were equated to be 1.58% related to the Support Vector Machine.

Another study concerning the prices of cryptocurrencies is that by Saad et al., 2020, through variable correlation analysis. This paper has evaluated transaction rates, hash rates, numbers of users, total bitcoins, and their correlations with prices. The done run analysis has shown the results of its dynamics in the cryptocurrency, guiding it directly to machine learning models: LR, RF, and GB. It was doing better than any former study using exclusively data of historical prices.

In other words, it is an excellent quality variable going into the making of a prediction model. From the discussion, it appears quite apparent that almost all the researchers confirm the primacy of first-class data quality in making a correct forecast for cryptocurrency pricing.

2.5 Hybrid machine and deep learning Methods

This section outlines some techniques and algorithms from the chosen research papers on hybrid machine and deep learning techniques that are used in the time-series forecasting application.

Hybrid models have been found to give great improvements in forecasting cryptocurrency prices. Patel et al. (2020) proposed an LSTM-GRU hybrid model that was put to use in predicting prices over different periods: one, three, and seven days for Litecoin and Monero. Their findings showed that the hybrid model outperformed the single LSTM method significantly. The best that could be realized was 2.06% MAPE in the three-day leading window ahead for Litecoin.

Livieris et al. (2021) developed a CNN-LSTM model in which features were taken from Bitcoin, Ethereum, and Ripple. Data for each cryptocurrency was given for processing separately in the model to avoid overfitting, and useful information was abstracted. In other words, the CNN-LSTM model was computationally more efficient compared to using a single CNN network.

Kristjanpoller and Minutolo (2018) proposed an ANN-GARCH model in dealing with Bitcoin price forecasting. The authors developed and tested twelve various models based on a mix of techniques and inputs. They found the optimal model to perform the best with a 1.64% mean absolute percentage error Exponential Generalized Autoregressive Conditional Heteroskedasticity model.

Altan et al. (2019) presented a hybrid model that combined the LSTM neural network with Empirical Wavelet Transform (EWT) decomposition and the Cuckoo Search (CS) algorithm. In comparison with a standalone LSTM, or the EWT-LSTM models, the EWT-LSTM-CS hybrid model delivered superior performance in forecasting prices of the Bitcoin, Litecoin, Digital Cash, and Ripple. The model effectively captured the non-linear characteristics of crypto-currencies.

The different hybrid models reviewed in this section have been found to be quite promising in enhancing cryptocurrency price forecasting. There are immense opportunities opened for future researchers on the topic, as the number of possible combinations of models and variables becomes huge.

2.6 Literature review summary

Various factors are influencing the performance of the forecasting models, and these shall be put into consideration while developing them. Among the most important aspects to be put into consideration is data quality. It is not all about applying high-level methods and techniques to be good at making predictions with regard to cryptocurrency. Accurate collection of data that will include variables impacting fluctuations in prices relating to cryptocurrency should be considered. As can be seen from the literatures reviewed, in the future, more researchers will consider macroeconomic factors and public sentiment to improve the model's performance. In 2020, Wang and Chen identified that the inclusion of variables representing public sentiment increases the performance of the models considerably. Nevertheless, the sentiment and opinion analysis techniques seem to be an area that has not yet been adopted by many studies.

Legislative uncertainty can also primarily influence the volatility in cryptocurrency. Sentiment and opinion analysis around proposed regulatory changes would possibly signal a price swing. The laws vary among countries, but influentially economic powerhouse countries like the United States would greatly affect cryptocurrency prices. Ferdiansyah et al. (2019) identify that the stock market—including cryptocurrencies—is under heavy influence from a lot of uncertainties and political factors.

Future studies should therefore consider hybrid models that integrate macroeconomic variables with sentiment analysis and public opinion, particularly those of countries with some appreciable influence on world economic matters.

3. Methodology

3.1 Methodology Strategy

Our methodology follows the deep learning system architecture as presented in figure 3.1. Deep learning represents a subfield of machine learning, aiming to imitate how humans gain a particular type of knowledge through experiences. The term 'Deep' represents using a neural network with more than three layers of depth. The network depth creates deep hierarchical representation learning where layers are stacked on top of each other. It is a multistage information distillation process where information is purified passing through several filters. The network learns data representation from the multi-stage sequence process.

As shown in figure 3.1, methodology was structured following a sequence of processes. First, bitcoin prices were collected using the yfinance library. Second, data preprocessing was done, which involved loading of data, dropping all irrelevant columns, splitting data into input features and target variables, and normalizing the data using both MinMaxScaler and StandardScaler. The dataset was then split into a training set and a test set, where the training set contained 90% of the total data, and another 10% remained in the test set. This information was then transformed into 3D tensors in a view to prepare it for the LSTM neural network.

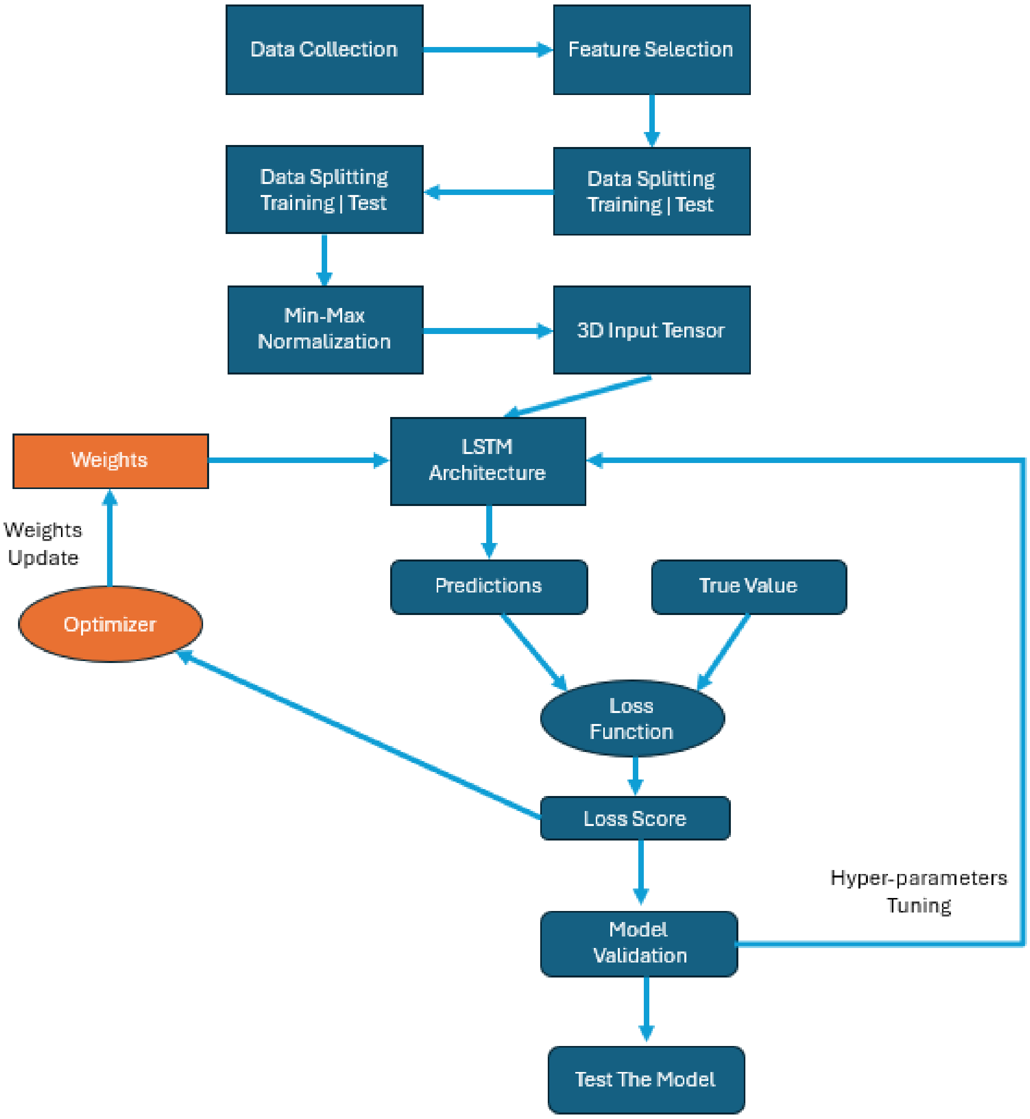


Figure 3.1: Deep learning System architecture

The selection of the right architecture in deep learning systems is paramount. In the conducted research study, a Long Short-term Memory neural network was used. This is an RNN type that is ideal for the processing of time series one at a time. The defined LSTM network, with specified layers and hidden states, was to handle the sequence data effectively.

It involved the observation and labeling of a large number of inputs, which were then matched to a deep sequence of data transformations, known as layers. The transformation executed on the inputs was done by the layer weights, also referred to as layer parameters. The learning process therefore entailed finding the layer weight values that enabled the network to map correctly the inputs and their associated labels. The reason why it is difficult to find the correct value for all weights is simply that a network can have many layers.

Making the LSTM network output the desired result, this algorithm first needs to observe and measure how far such output was from the real value. This measurement was done through a specially defined function, one that makes a comparison of distances between the forecasts and the real value, so-called loss score. It used the loss score for the response signal to be able to adjust the weights values in such a manner that the algorithm will strive towards reducing the loss score. An optimizer operating with a gradient-descent algorithm was applied to make these adjustments. The gradient of the loss concerning model parameters is computed to find the downhill direction, and then the weights are moved in small steps in the opposite direction from the gradient, letting the loss reduce a little each iteration.

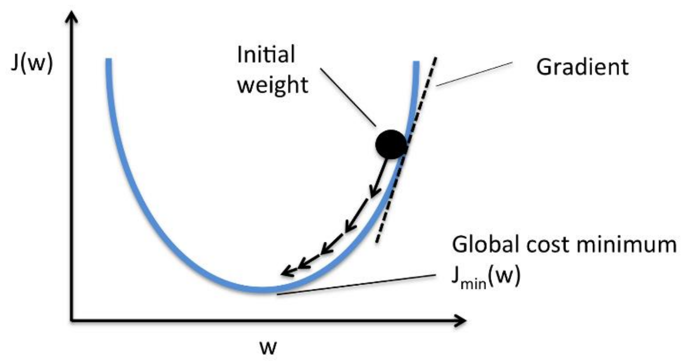


Figure 3.2: Weights optimizer

The optimizer works by randomly initializing the weights and repetitively adjusting them with small steps until it converges to a value close to the global minimum. This is attained using the gradient of loss and the learning rate hyperparameter. The learning rate is what dictates how fast gradient descent happens, thus the importance of having a sensible value for this hyperparameter. If the learning rate is too small, then it will need to go through many iterations, and there is a danger that the loss value will get stuck in a local minimum. If the learning rate is too large, the loss value may overshoot the global minimum and jump to completely random locations on the loss curve. The gradient descent algorithm measures the local gradient of the loss value with respect to the weights, and then it follows that very direction to get a greater gradient descent. If the gradient equals zero, it means that the minimum has already been reached. The gradient describes how much the loss value changes when the weights are tipped a little. The process is repeated until the minimum has been found.

There are several hyper-parameters in the LSTM network that were tuned in order to enhance the performance of the algorithm's predictions. Such response signal was taken from the model validation error, while the LSTM hyper-parameters were adjusted in a manual fine-tuning strategy in a way that lets the algorithm minimize the validation error.

3.2 Data Collection

Our methodology follows the deep learning system architecture as presented in the figure. Deep learning represents a subfield of machine learning that aims to imitate how humans gain specific knowledge from experiences. Now, the term 'Deep' represents using a neural network with more than three layers of depth. It is the depth of the network that thus creates deep hierarchical representation learning where the layers are stacked on top of each other. It is a multistage information distillation process whereby information is purified through the passing of several filters. The network learns data representation from the multi-stage sequence process. Analysis of variables majorly used by articles included in the literature review drove the data collection process. Yfinance was the Python application program interface used to download data from the website Yahoo Finance. We retrieved the prices for Bitcoin -USD and Ethereum - USD from March 3rd, 2023, to March 3rd, 2024.

3.3 Features Description

Perhaps the most important factor in deriving optimal performance from any deep learning prediction algorithm is data quality. Thus, feature selection was informed through the literature, which had noted that macroeconomic variables contribute immensely to model prediction performances. The features used within this research study are described in Table 3.1.

Figure 3.1: Features Description

Features	Description
Close	Bitcoin closing price in USD
Open	Bitcoin opening price in USD
High	Bitcoin highest price of the day in USD
Low	Bitcoin lowest price of the day in USD
Volume	Bitcoin total transactions volume of the day

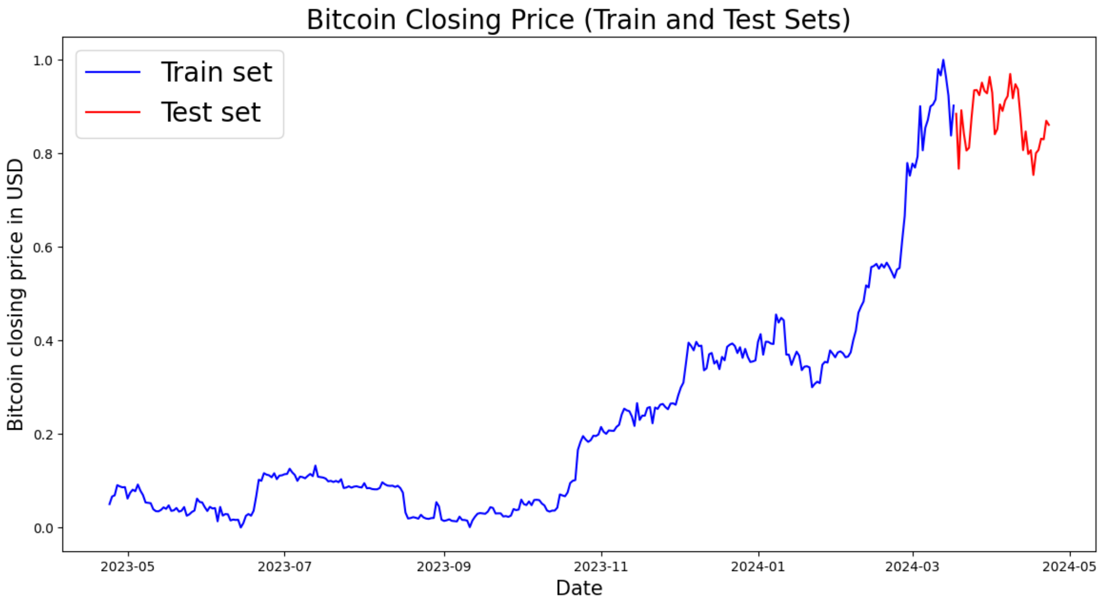
3.4 Feature Selection

In this work, 4 different features are collected to be initially used as input to deep learning models. In the feature selection process, it retrieves an optimal subset from the feature set to increase model performance. It was implemented by using a wrapper forward selection method that basically starts with one feature, and iteratively adds more features to the current set as long as improvement in the model's performance exists. After applying this method, it was found that the models predicting the future prices of Bitcoin with the closing price alone as the input feature are less noisy, computationally less expensive, and have lower forecasting errors. Thus, based on these findings, the final models used only the Bitcoin closing price as the input feature.

3.5 Train/Test Data

The dataset was then split: 90% for training and 10% for testing. Data from April 24th, 2023, to March 24th, 2024, was used for training, and from March 25th, 2024, to April 24th, 2024, data was set aside for testing. Annexed are the visualizations. This is to train and test the model on unseen data. These split percentages were chosen after some testing to ensure the model works fine, so the lowest test prediction error indicated the optimal performance.

Figure 3.3: Train and test bitcoin closing price (USD per bitcoin)



3.6 Data Pre-processing

This section details all the operations done in the phase for data pre-processing. First, Bitcoin's data was gathered using the yfinance library. After that, the dataset was split into a training subset with 90% and another testing subset with 10%. Finally, data normalization is done using MinMaxScaler to make sure that all values are on the same scale. Finally, data was structured into a 3D tensor format that would feed into an LSTM neural network input. More precisely, it created sequences of 10 time steps as input and 1 time step as the label.

3.6.1. Feature Scaling

Feature scaling is among the most important data transformations, to be done, as deep learning algorithms normally do not work well when features of the system have scales that differ from one another. In that regard, the Min-Max normalization technique is applied for scaling the data within the range of 0 and 1. Result is the subtraction of each observed value and division by the result of the minimum value from the range against the range, which is the difference between minimum and maximum values, so that each value remains common in the dataset.

3.6.2. Data Representation of Neural Networks

We utilized a rank-3 tensor for preparing data for deep learning algorithms. People can think of this visually like a cube with compartments; its axes denote samples, the number of time steps, and features. Tensors are basic data structures by which deep learning systems use and generalize matrices—rank-2 tensors—to handle an arbitrary number of dimensions. They are designed for somewhat like a container to hold and manipulate data in a system.

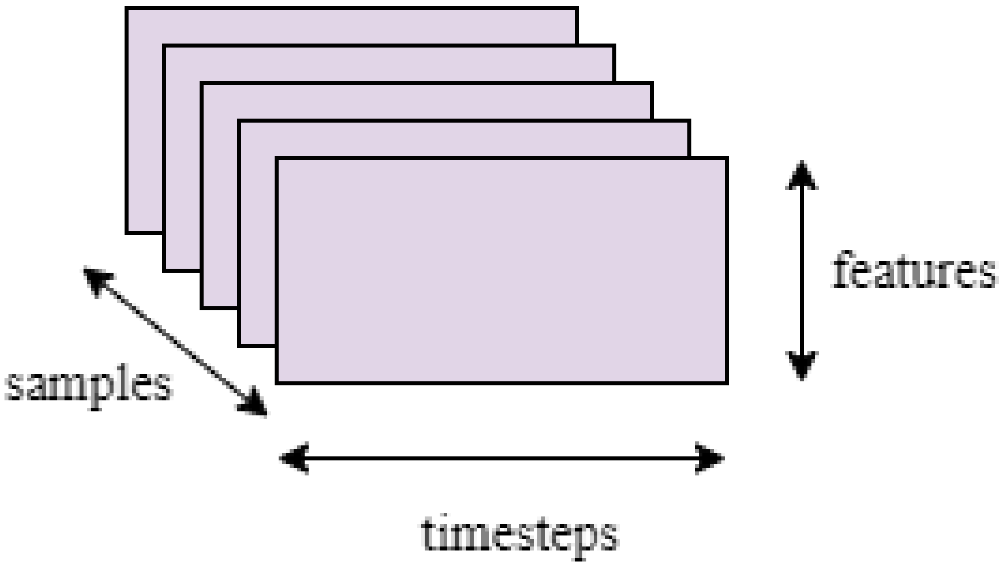


Figure 3.4: Rank-3 timeseries data tensor

Consider our approach: a 10 time-step sequence where each step is the closing price of bitcoin for a day. Such structure allowed us to build a rolling window, turning the dataset into a supervised learning problem with an input and label. More specifically, the last 10 days' prices of bitcoin served as input to predict the closing price of bitcoin for the next day, which would be the label.

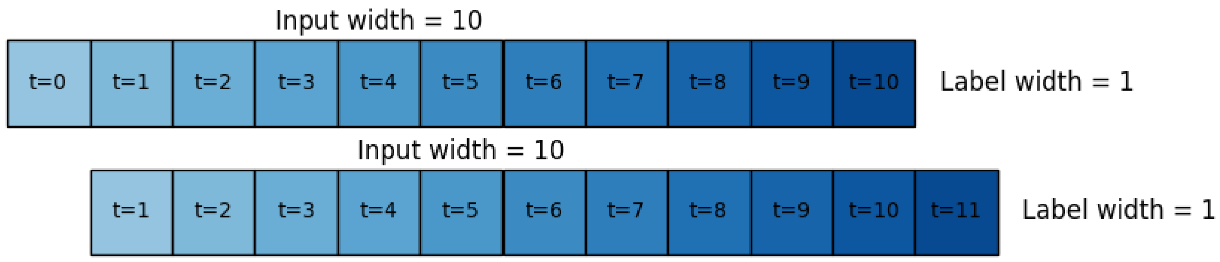


Figure 3.4: Supervised learning rolling window

3.7 Modelling

This section presents the LSTM and algorithm and the architecture of the final model's implementation.

3.7.1. LSTM network

Most deep learning neural networks have no built-in capability for memory and thus all time steps of input are processed as one chunk of data. This strategy maps inputs to labels without modeling temporal patterns that exist in the sequences. RNNs were developed to take care of this limitation by processing time dependencies in a better way. Although RNNs do quite well on the handling of short-term dependencies, they don't do as well on handling long-term dependencies. This limitation led Hochreiter and Schmidhuber to develop, in 1997, Long Short-Term Memory networks. Actually, LSTMs are designed especially for learning short-term and long-term dependencies. Consequently, they are particularly efficient with sequential data, hence for time series analysis. As seen in Figure 3.6, an LSTM network includes chain modules which repeat, each corresponding to a time step in the sequence.

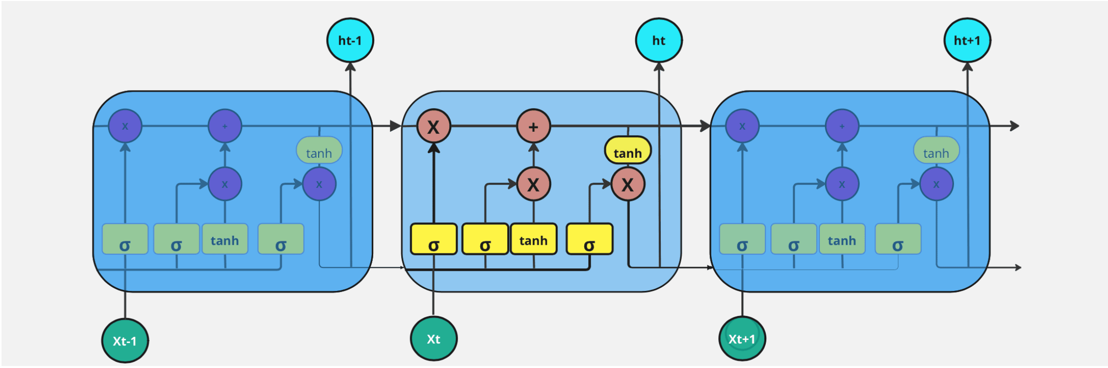


Figure 3.6: LSTM chain modules

Probably the most important characteristic feature of the LSTM network is the cell state (C_t), a vector flowing through the complete chain of cell modules, as illustrated by Figure 3.7. The long-term information and patterns are held in this cell state, whereby minimal linear interactions are involved in their storing and handling dependencies.

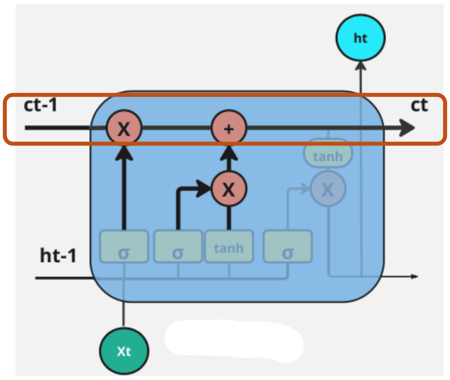


Figure 3.7: LSTM cell state

In the LSTM network module, there are three gates, and every gate is an instantiated sigmoid neural network layer. These three kinds of filters that the gates implement control what information should be added to or removed from cell state (C_t) (as shown in Figure 3.8).

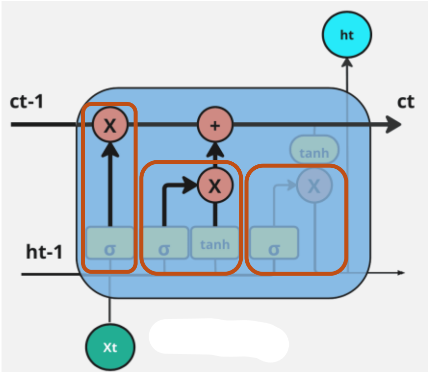


Figure 3.8: LSTM gates

The first operation in an LSTM cell is to decide what parts of the information stored in the previous time step's cell state, (C_{t-1}), are to be thrown away. This is done by utilizing what's called a forget gate — that is, a sigmoid layer — as part of the process. The forget gate takes the concatenated input of the current time step, (x_t), and hidden vector of the previous time step, (h_{t-1}), multiply these two with the weight matrix, (W_f), and adds the bias term, (b_f). It then applies the sigmoid function as shown in Figure 3.9, Equation 3.4. It produces an output, value per component of the cell state, thus between 0 and 1, so that 1 for (C_{t-1}) means "keep all information," and 0 means "dump all information".

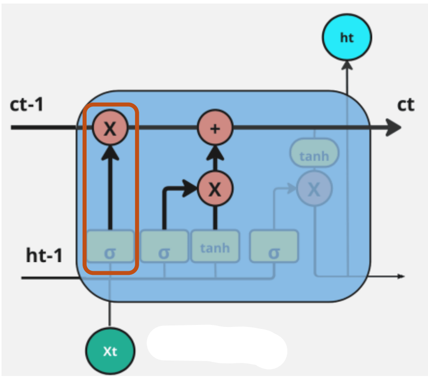


Figure 3.9: LSTM forget gate

$$ft = \sigma(W_f.[h_{t-1}, x_t] + b_f) \tag{3.4}$$

The second step of an LSTM cell is to decide what new information is added into the cell state (C_t). This operation is completed by a two-stage process. First, it contains the "input gate", (i_t), which is a sigmoid layer conditioning and thus selecting the values to update. In this case, the input gate consolidates the current time step's input data (x_t), concatenated with the last time step's hidden vector (h_{t-1}), using a learned weight matrix with an added bias term before applying the sigmoid(b_i) (equation 3.5). In the second part, a hyperbolic tangent layer (tanh), produces a vector of candidate values (C'_t) to the cell state (C_t). In particular, this is accomplished by concatenating the input data for the current time step, (x_t), with the hidden vector from the prior time step, (h_{t-1}), multiplying by the weight matrix, (W_c), adding the bias term, (b_c) and taking the hyperbolic tangent function, (tanh), of this expression (equation 3.6) (figure 3.10).

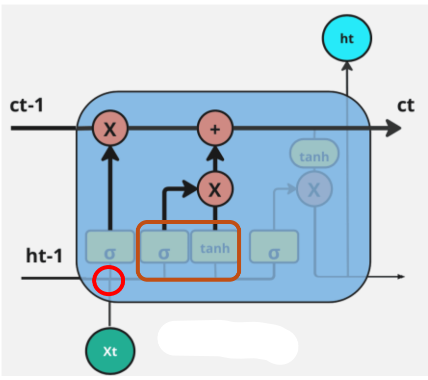


Figure 3.10: LSTM input gate

$$i_t = \sigma(W_i \,.[h_{t-1}, x_t\,] + b_i) \tag{3.5}$$

$$C'_t = \tanh\,(W_c \,.[h_{t-1}, x_t\,] + b_c) \tag{3.6}$$

The third step involves incorporating the results from these two stages in determining which information has to be brought forward to the current cell state (C_t). This is accomplished through the multiplication of the candidate vector of values, (C'_t), with the input gate, (i_t), which filters out information and makes decisions regarding which elements from the input data of the current time step, (x_t), and the previous hidden vector, (h_{t-1}), should be important and integrated into the new cell state (C_t). The process is illustrated in Figure 3.11.

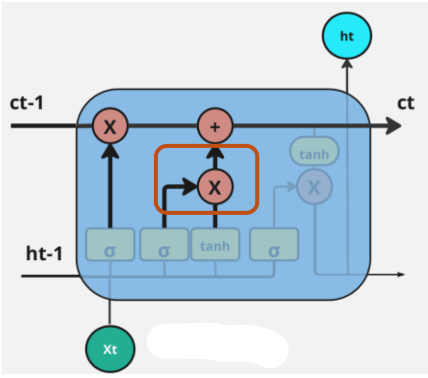


Figure 3.11: LSTM input gate

The fourth step updates the cell state (C_t) with the forget gate (f_t)and the input gate (i_t). This is simply a matter of adding, separately, the information to be forgotten from the previous time step, ($C_t * f_t$), with the new information to be added, ($C_t * i_t$). This sum then updates the old cell state, (C_{t-1}), to the new cell state, (C_t) as shown in figure 3.12 and in equations 3.7 and 3.8.

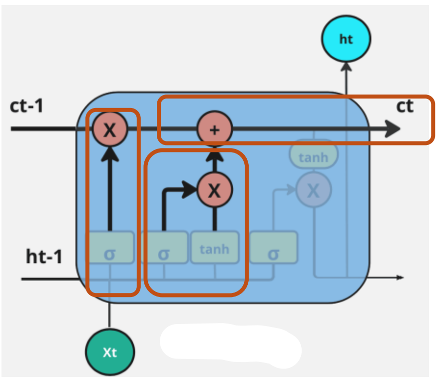


Figure 3.12: Update LSTM cell state

$$C_t = C_t^f + C_t^i \tag{3.7}$$

$$C_t = f_t * C_{t-1} + i_t * C'_t \tag{3.8}$$

The last step a cell does is to determine the output for this current time step. An output gate, (o_t), decides the output through a sigmoid layer that filters the output and the hidden state for this current time step (h_t). The output gate, (o_t) can be obtained by applying the sigmoid function on the concatenation of the current time step input data, (x_t) with the previous time step's hidden state, (h_{t-1}), by multiplying it with the weight matrix (W_o), which is added to the bias term (b_o) as shown in equation 3.9. The current cell state, (C_t) , is passed through the hyperbolic tangent, (tanh), function and multiplied by the output gate, (o_t). This result forms the hidden state for the current time step, (h_t) , and the output to be passed to the dense layer for prediction by the LSTM network as shown in figure 3.13 and described in equation 3.10.

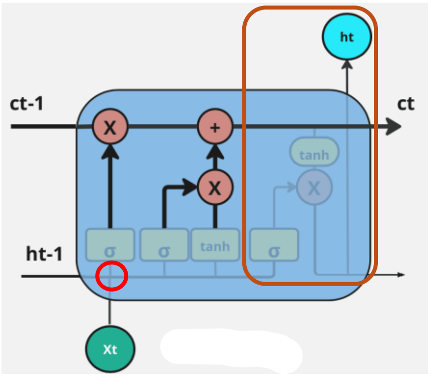


Figure 3.13: LSTM output gate

$$o_t = \sigma(W_o\,[h_{t-1}, x_t] + b_o) \tag{3.9}$$

$$h_t = o_t * \tanh(C_t) \tag{3.10}$$

3.7.2. Final models Implementation

This section presents the implementation of the final developed LSTM model in detail.

3.7.2.1. LSTM architecture

We implemented an LSTM model in Python using its PyTorch library. On the first input layer, it took a prepared rank-3 tensor during the data preprocessing phase, of the shape :

[{batch size} = 32, {time steps} = 10, {number of features} = 4]

These 32 batches enabled us to work out the prediction error at each iteration and hence update the weights of this LSTM architecture to reduce such an error. As mentioned earlier, we used 10 time steps, which refers to ten days as an input for the prediction of the closing price of the following day for Bitcoin.

The second layer of the LSTM architecture consisted of 2 hidden units. In the output layer for predicting the next day's closing price of Bitcoin, there was a dense layer of 1 unit, corresponding to a 1-day forecast. The model was trained using 90% of the data and tested on the remaining 10%.

For preprocessing, MinMaxScaler will be used to normalize the close price of Bitcoin within a range of 0 and 1. Afterwards, this data will be prepared where sequences of 10 time steps are created for the input and 1 time step for the output. Then, training and test sets will be created by splitting this normalized data in the ratio 90% for training and 10% for testing.

For regularization purposes, the LSTM model architecture includes a Dropout layer. This was set so that, during training, dropout would randomly exclude 20% of the layer's output features. This will avoid overfitting of the model to the training data and hence make it based on the validation and testing data. This regularization will lead to a more general model with a smoother prediction curve and better performance.

The model trained for more than 1000 epochs. A system of early stopping ensured that the process stops if validation loss does not improve for more than 30 epochs, so that the best model obtained during training is saved. This helped stop training as soon as possible when the model started overfitting.

The mean squared error loss function provided the response signal, which was then used in tuning the value of the weights to ensure a low loss score. This adjustment in weights was done through the Adam optimizer, working by modifying the weights in the direction that minimizes the prediction error. The Adam algorithm – Kingma & Ba, 2014 – is based on the stochastic gradient descent method.

The learning curves were plotted for the training and validation loss during the training process. The validation and training losses decreased until the curves were very flat and maintained a minimum gap between the two, hence a good model fit with respect to the training and validation data.

3.7.2.2. Hyper-parameters tuning and regularization

One of the most critical tasks in the building and refinement of the LSTM model was that of hyperparameter tuning. This was for the achievement of good predictive performance. The goal was to develop a model with high generalization capability so that it performed well not only on training data but also on unseen validation data. In doing this, various configurations were tried out with special emphasis on the following hyperparameters:

- Number of Layers
- Number of Units per Layer
- Batch Size
- Optimizer and Learning Rate
- Regularization (Dropout)

The final model was achieved by continuously adjusting the number of units, the learning rate, the optimizer, and the dropout percentage, and by rigorously validating the model at each step. In each step by iteration, comprehensive model validation has been done to make sure that the model developed realizes the best performance for the task at hand.

3.8 Evaluation of deep learning models

To ensure that this model performed well, its results must be compared with the baseline model represented by the 20-day moving average and with those from other studies within the review. To this effect, the LSTM model was trained using the dataset; that is, testing involved making some predictions using the test data. These would then be converted back to the original dollar scale to put them at par with the observed values.

In total, only one performance metric was used, which helped to observe the model's performance holistically. Mean Absolute Percentage Error is a measure for which calculation was done for the prediction accuracy. These evaluations show how well the model would work in forecasting as compared to the 20-day moving average, a much simpler model—thereby showing potential improvements of an LSTM capturing the trends and making more accurate predictions.

3.8.1. Mean Absolute Percentage Error

The model performance metric chosen was the Mean Absolute Percentage Error (MAPE), which describes the accuracy of a prediction in percentage terms over its actual. It gives a benchmark criterion through which one can compare and contrast the effectiveness of the model. Perception therefore is very good from this value because the magnitude of the percentage is small. Therefore, this metric indicates to what extent the values of the model predictions are apart from the observed ones; thus, a smaller magnitude of the percentage is better.

$$MAPE = \frac{100}{n} \sum_{t=1}^{n} \left| \frac{y_t - \hat{y}_t}{y_t} \right| \tag{3.11}$$

Where n is the number of observations used for testing, y is the observed value, $\hat{y}$ is the predicted value and t is the time step.

4. Data and Results

This is where the data analysis, the basic descriptive statistics, and the findings from the experiment conducted for feature selection, hyper-parameter tuning, and dropout regularization are discussed. All experiments were trained on the dataset for training and tested on the test dataset. The configurations of the LSTM architecture used are as described in the methodology.

4.1 Descriptive statistics

The median Bitcoin closing price, within the range of collected data, was approximately 30,171.23 USD per Bitcoin. This value is somewhat below the mean, which stands at 35,878.65 USD per Bitcoin. The difference is thus in the fact that while prices for Bitcoin have generally been high, times have occurred where prices were lower to drag down the median from the mean.

The data collected on the close price of Bitcoin showed large dispersion. In the statistical analysis, it is observed that there is a huge difference between the minimum and maximum levels, at 25,124.68 USD and 73,083.50 USD, respectively, against a standard deviation of 11,561.68 USD, portraying an extreme change in the value of Bitcoin within the period under consideration.

Table 4.1: Statistics

Date	Open	High	Low	Close	Adj Close	Volume
Mar 1, 2024	61,168.06	73,750.07	59,323.91	71,333.65	71,333.65	1,446,417,844,950
Feb 1, 2024	42,569.76	63,913.13	41,879.19	61,198.38	61,198.38	830,721,862,621
Jan 1, 2024	42,280.23	48,969.37	38,521.89	42,582.61	42,582.61	825,918,941,347
Dec 1, 2023	37,718.01	44,705.52	37,629.36	42,265.19	42,265.19	721,704,910,480
Nov 1, 2023	34,657.27	38,415.34	34,133.44	37,712.75	37,712.75	570,863,267,380
Oct 1, 2023	26,967.40	35,150.43	26,558.32	34,667.78	34,667.78	476,425,634,860
Sep 1, 2023	25,934.02	27,488.76	24,930.30	26,967.92	26,967.92	337,637,313,742
Aug 1, 2023	29,230.87	30,176.80	25,409.11	25,931.47	25,931.47	437,724,169,499
Jul 1, 2023	30,471.85	31,814.52	28,934.29	29,230.11	29,230.11	382,224,489,090
Jun 1, 2023	27,218.41	31,389.54	24,797.17	30,477.25	30,477.25	481,734,214,225
May 1, 2023	29,227.10	29,820.13	25,878.43	27,219.66	27,219.66	443,473,015,479
Apr 1, 2023	28,473.33	31,005.61	27,070.85	29,268.81	29,268.81	511,540,319,004

In the visual analysis below, excluded here for brevity, one can trace the trends in Bitcoin's price over the period. For example, there could be periods of rapid increases or decreases, like those identified in the history of the cryptocurrency market.

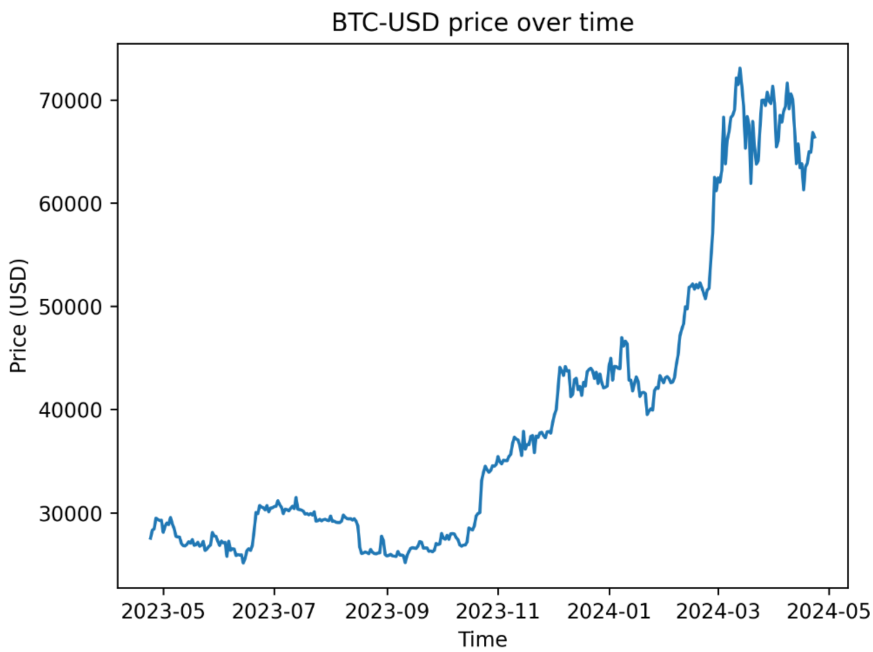


Figure 4.1: Evolution of Bitcoin closing price (USD per Bitcoin)

4.2 Feature Selection

Models developed on a single feature, the closing price of Bitcoin, are associated with less noise, reduced forecast errors, and minimized complexity and execution times. Tables 4.3 and 4.4 indicate experiments performed by various combinations of input features, such as the closing price, opening price, and the highest price of the day, to feed LSTM-based deep learning models.

Multiple-feature experiments using closing, opening, and the highest price of Bitcoin in a day created models of higher accuracy measured by forecast errors. Using only the closing price as the input feature created a less noisy, simpler, and faster model to run, as shown in Figure 4.2.

Table 4.3: LSTM models performance with different set of features

LSTM	
Feature Combination	**MAPE (%)**
Close / Open	4.1
Close / Open / High	6.6
Close / Open / High / Low	6.1
Close / Open / High / Volume	4.0

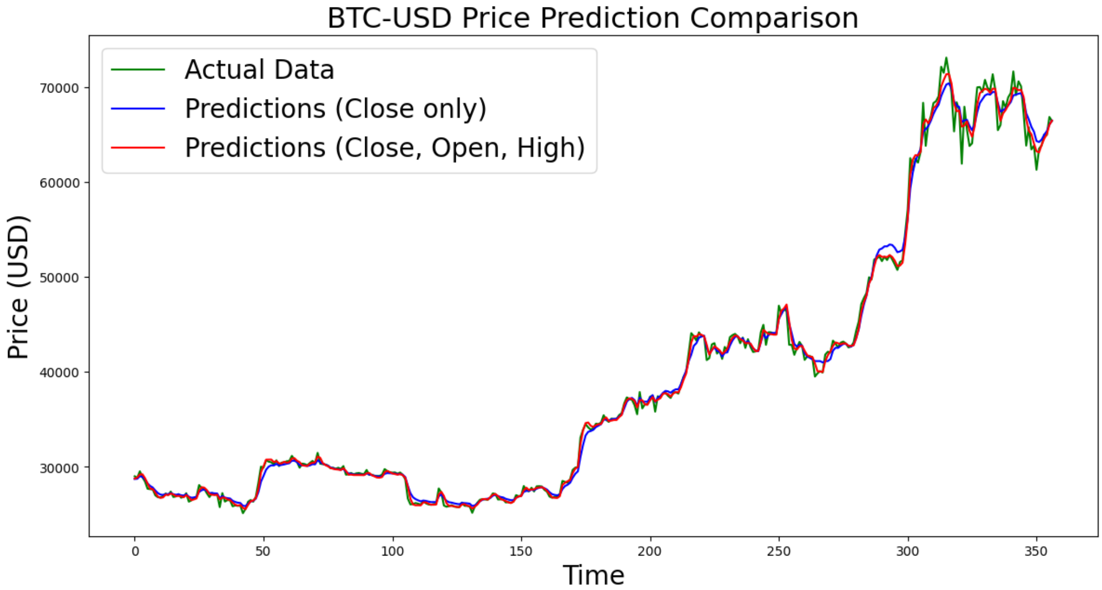


Figure 4.2: Actual and LSTM predicted bitcoin closing price with different set of input features

4.3 Hyper-parameters tuning

This chapter gives the result of hyperparameter tuning by grid search in the models for predicting the close price of Bitcoin with LSTMs. As this is one source through which other researchers will be able to deduce effective approaches toward attaining optimal hyperparameters, the commitment shall be toward ensuring that highly accurate levels are obtained. Also, the prediction errors in the subsequent sections originate from implemented experiments and analyses on the LSTM models, as earlier explained in the methodology of this work.

4.3.1. Number of hidden layers

The best performance to predict the closing price of Bitcoin belonged to a single LSTM layer. As presented in Table 4.5, although the lowest prediction error for LSTM model came out with a 3-layer model, the single-layer configuration generalizes better to new data. This provided a more stable and smoother forecast, with reduced noise, as evidenced by clearer and more consistent lines of forecasts in Figure 4.3.

Table 4.5: LSTM models performance with different number of layers

LSTM	
Number of layers	MAPE (%)
1	2.1
2	2.62
3	1.04

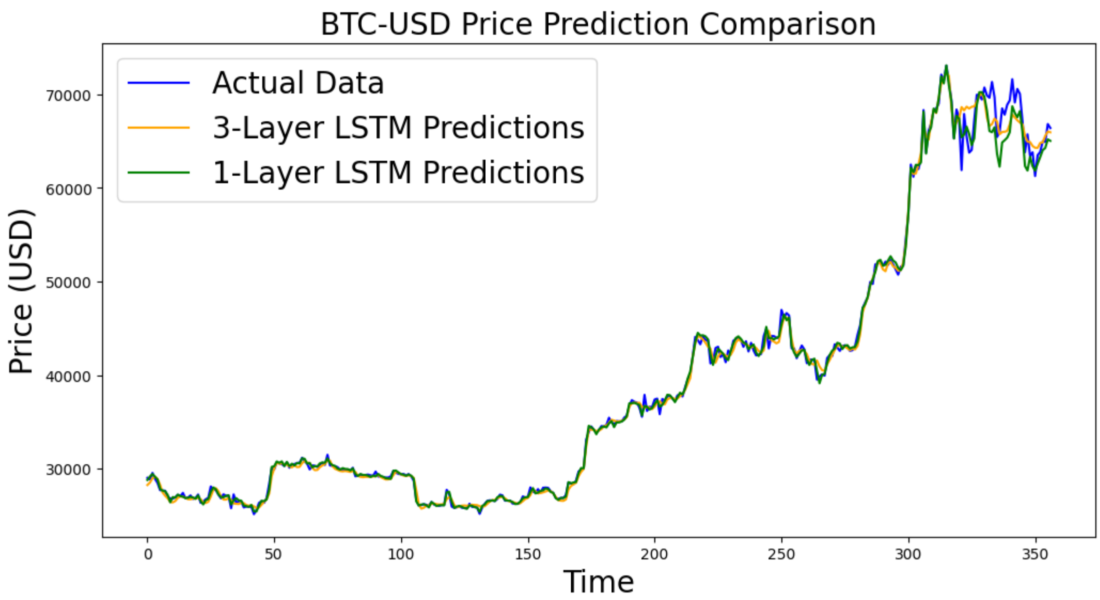


Figure 4.3: Actual and LSTM predicted bitcoin closing price with different set of layers

4.3.2. Number of units in the LSTM hidden layer

The number of units was a critical factor for building an appropriate Bitcoin close price model. Table 4.6 shows experiments regarding various numbers of units in the LSTM hidden layers. In results, the models with 2 and 256 units performed remarkably better that the others. Furthermore, based on the conducted analysis, it was able to be found that while the number of units was going up, the training process turned out to be slower.

Table 4.4: LSTM models performance with different number of units

LSTM	
Number of units	MAPE (%)
2	0.07
4	0.91
8	0.56
16	1.19
32	0.52
64	0.41
128	0.25
256	0.09
512	0.79

4.3.3. Learning rate

The learning rate was very critical to good forecasting performance. Table 4.9 shows the validation performance of LSTM model when different learning rates are used with the Adam optimizer. From the results, the prediction accuracy was best when the model had a learning rate of 0.01. The learning rate of 0.0001 extremely degraded the outcome with the LSTM model. Specifically, 0.0001 learning rate was too low. Then, the 0.1 learning rate, was too large; therefore, the loss value overshot and shook around the global minimum in an erratic manner.

Table 4.5: LSTM models performance with different learning rates

LSTM	
Learning Rate	MAPE (%)
0.1	0.48
0.01	0.04
0.001	0.83
0.0001	7.18

4.4 Dropout regularization

Dropout regularization assisted us to get a more regular, simplified, and generalized model with a smoother prediction curve and performance improvement on the validation data. Table 4.6 and Figure 4.10 illustrate experiments with and without dropout where the layer output features during training were excluded from 20% to 60%. The visualizations show that in case of a lower prediction error when not using the dropout; however, applying the 20% dropout rate will result in a more regular, simpler, smoother, and generalized prediction of the Bitcoin closing price.

Table 4.6: LSTM models performance with different dropout percentage

LSTM	
Dropout Percetage	**MAPE (%)**
0%	0.33
10%	0.31
20%	0.07
60%	1.13

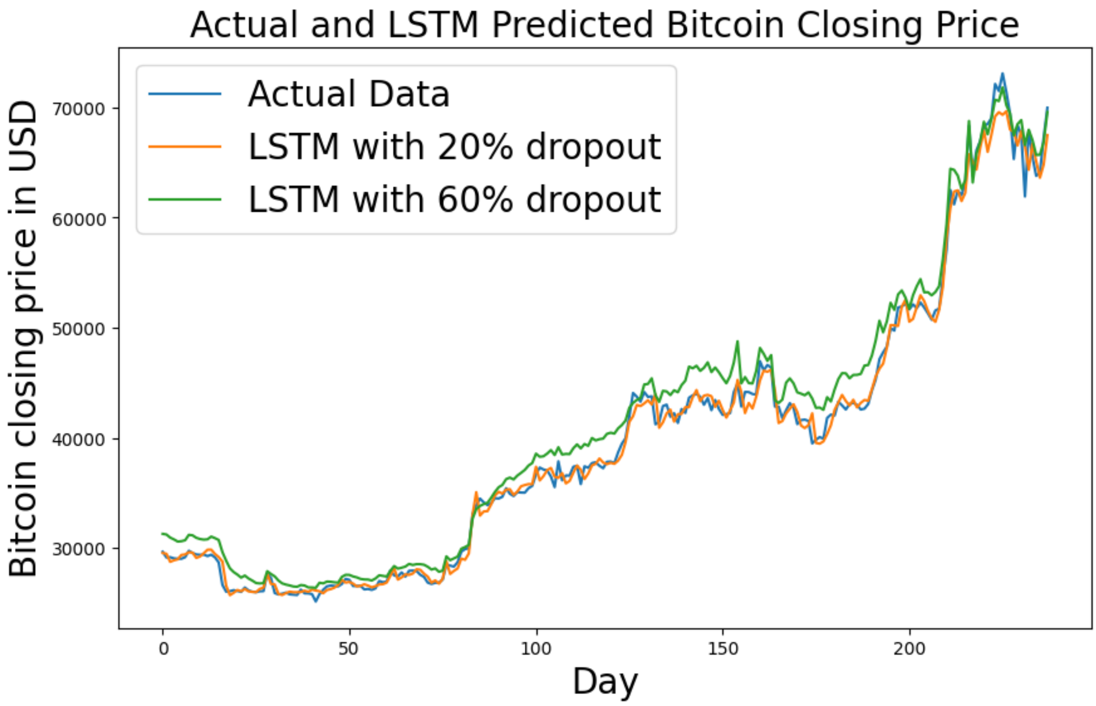


Figure 4.4: Actual and LSTM predicted bitcoin closing price with 20% and 60% dropout percentage

4.5 Final LSTM model evaluation

This part shows the final result from the evaluation executed by the LSTM model for the test dataset. The entire process of training and testing of the model is run many times for informative statistics on performance. This is done only when a model has been fine-tuned and validated with validation data. As such, the prediction errors discussed here are a result of the final LSTM model configuration described within the methodology. The LSTM model performed very well in predicting the closing price of Bitcoin, returning an MAPE of 0.07%. In addition, the training phase of this LSTM model was very efficient and required far less time to complete than other traditional ways of training.

The dropout regularization tested at 0%, 10%, 20%, and 60% showed that in case of no use of dropout, this resulted in a lower prediction error. Also, the use of a 20% dropout resulted in a more regular and smoother generic prediction curve. This proves that dropout regularization is very important for generalization in a model and obviously comes with a smoother prediction curve at 20% dropout.

Table 4.7: Final LSTM models performance

LSTM	
Time (sec)	MAPE (%)
8.5	0.07

In figure 4.5 the plot presents the actual and predicted bitcoin closing prices for LSTM. The orange line is the result of the bitcoin closing price prediction, and the blue line is the actual bitcoin closing prices from the test data.

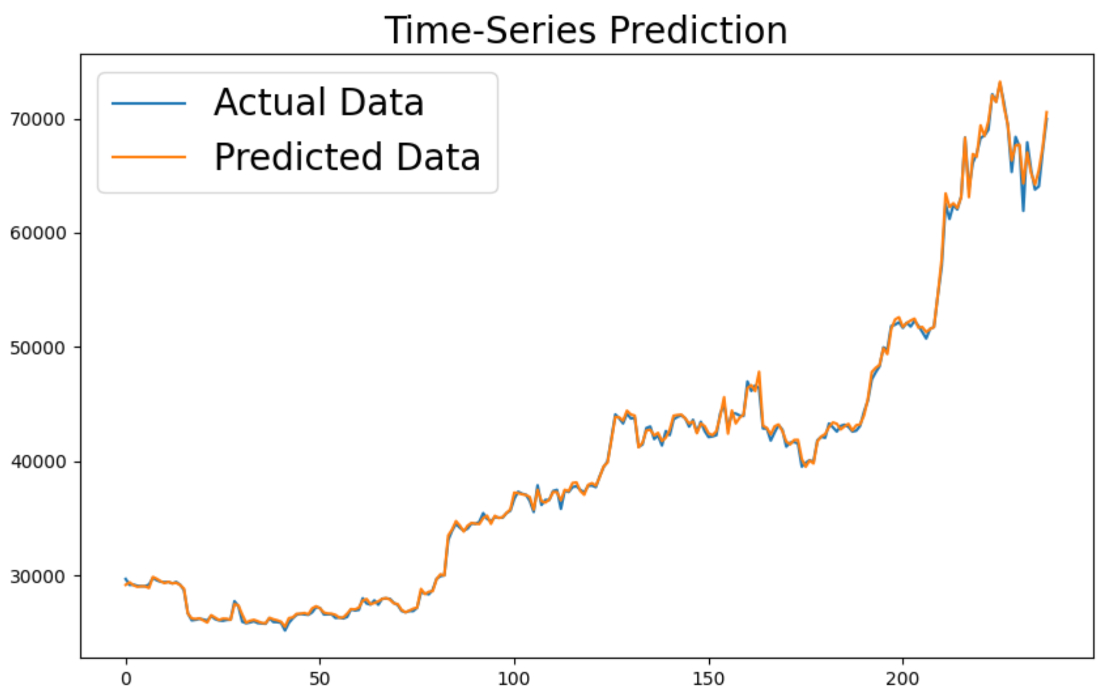


Table 4.5: Actual and predicted LSTM bitcoin closing price

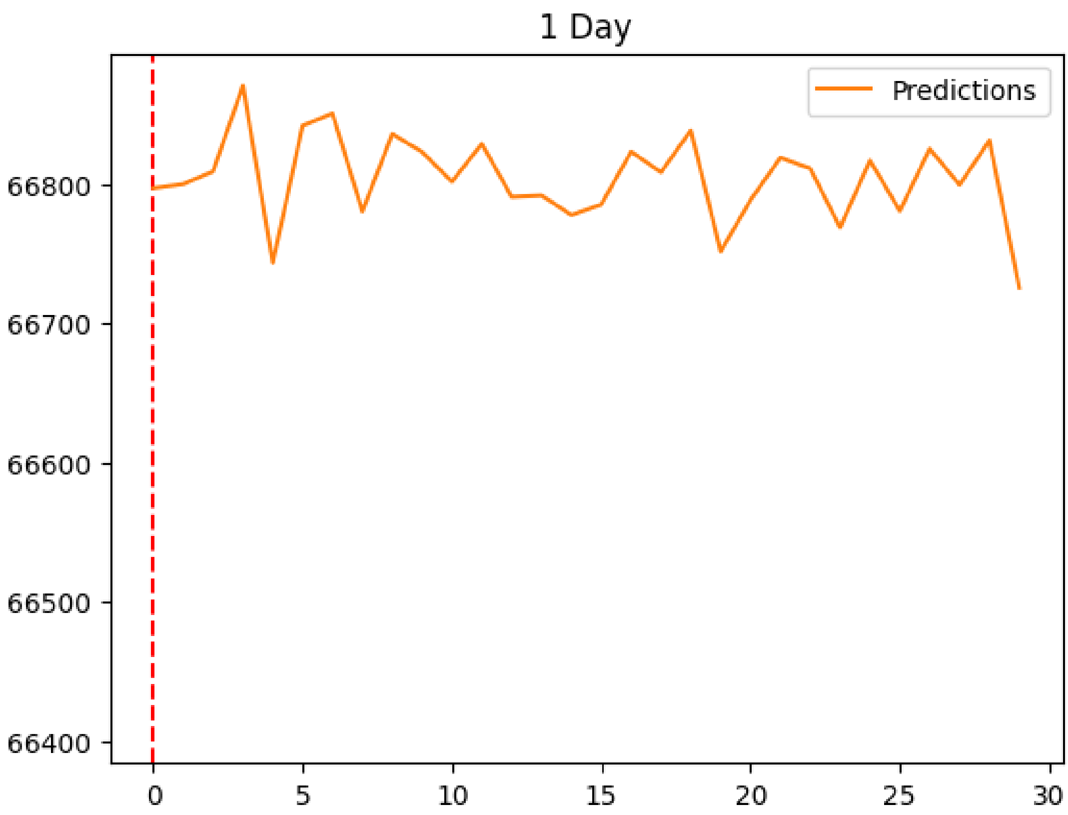


Table 4.6: Next-day LSTM bitcoin closing price prediction

5. Discussion

5.1 Key Findings

This dissertation presents works that provide evidence that LSTM models are efficient ways to predict Bitcoin closing prices. Besides, we could confirm that the use of dropout regularization is very effective to improve the models' performance. There was no evidence found to say the use of micro and macroeconomic variables bettered the performance.

5.2 Study Limitations

Like every other study, the current study also carries some limitations. Mostly, it was noticed that the economic variables collected for this research have very little significance or close to insignificance, hence making the signal more noisy that increases the prediction errors and enhances complexity with longer execution time; therefore, the final models were built only by using Bitcoin closing prices. With respect to the analysis of public sentiment on cryptocurrency investment, this aspect has been removed from the scope of this dissertation for the master's program due to time limitations.

5.3 Limitations from studies

Several limitations emerged from the studies reviewed. One major issue is that most of the studies only focus on enhancing performance by using more complex models and techniques and do not necessarily stress the collection of appropriate information to improve model accuracy. It was also realized that most studies did not handle the complexity and non-stationarity aspects in cryptocurrency time series data very well. Few of them used methods of differentiation or filtering to smooth the high volatility of prices of cryptocurrencies; this would have benefited both statistical and deep learning methods. Moreover, public sentiment and policies and regulations related to digital currencies were also ignored by the majority of the studies.

5.4 Strengths

Even though research had its limitations, it remained important due to some of these following key reasons. These were one of the pioneering efforts that used dropout regularization together with the LSTM models while predicting the prices of Bitcoin. Dropout was used so that overfit could be avoided in the predictions from the models; otherwise, the overfitting of the models might be on the higher side, and generalization too could be improved. The models developed in this research achieved a low prediction error, relatively of the order of the studies reviewed, on test data. These models could predict quite accurately the price and trend of Bitcoin, without major noise, and with really smooth forecast lines. It is therefore conclusive that the applied strategy in this methodology, through deep description of the deep learning architecture, serves as a contribution towards further research.

5.5 Incidental observation

Although an exploration in that direction lay beyond the scope of the current work, some side observations were beginning to emerge. It was observed that correct parameterization had a very big part to play in a correctly well-performing model. The number of hidden layers, number of units per layer, batch size, the choice of optimizer, learning rates, etc., play highly important roles in the accuracy pertaining to the predictive models. The best model-scaling technique for underfitting and overfitting at the edge is possibly that of applying dropout as regularization, which proved very good at predicting Bitcoin close prices. It was very exhaustive and time-consuming, but this is what hyperparameter tuning is all about. Models will be put together based on heuristics that will give strong results.

5.6 Comparison with deep learning models included in the review

Table 5.1 compares the models developed in this study with other next-day Bitcoin deep learning models included in the review.

From the entire studies in the review, the models predicted in this study got best for LSTM single models and the best among all deep learning studies when regarding MAPE. For a Bitcoin investor, how the price will trend in the future is more valuable than knowing the exact price in the future. Several studies predicted Bitcoin prices within a very close correlation to the price until the last day and ultimately resulted in high noise predictions. The research solved this issue by using dropout regularization and careful parameterization for avoiding overfitting, thus obtaining a smoother predicted line with much less noise and lower prediction errors.

Radityo et al. (2018) developed the two deep learning models that scored the lowest in terms of the MAPE under review and had scored the MAPE of 1.88% and 1.998% by using the genetic algorithm backpropagation neural network and the backpropagation neural network, respectively. The essence in the development of these models was the strong driving of the Exponential Moving Averages, 12-day Rolling Window, Volume, High, Low, and Close Prices in the prediction of Bitcoins. The results for the ANN and RNN models developed by Mallqui and Fernandes (2019) were 3.06% and 3.36%, respectively—very good compared to most of the deep learning models reviewed. Their success is related to something called Correlation-based Feature Subset selection, which estimates the worth of variable subsets based on the predictive power of each attribute and the level of redundancy that exists between them.

As can be seen from Table 5.1, though algorithm selection is quite important, there are other aspects comprising the predictive performance of the built algorithm.

These include data processing, feature selection, feature generation, and optimization of hyper-parameters.

Table 5.1: Deep learning models' errors included in the review

Nº	Author (year)	Cryptocurrency	Interval data	Method	MAPE (%)
This study		Bitcoin	1-Day	LSTM	0.07
4	Ferdiansyah et al. (2019)	Bitcoin	1-Day	LSTM	7.2
5	Livieris et al. (2021)	Bitcoin	1-Day	LSTM	9.0
6	Rizwan et al. (2019)	Bitcoin	1-Day	GRU	9.0
				LSTM	
8	Lahmiri and Bekiros (2019)	Bitcoin	1-Day	DLNN	9.0
				GRNN	
11	Altan et al. (2019)	Bitcoin	1-Day	LSTM	9.59
				EWT-LSTM	6.14
				EWT-LSTM-CS	3.55
15	Liu et al. (2021)	Bitcoin	1-Day	BPNN	37.36
				SDAE	10.19
16	Tan and Kashef (2019)	Bitcoin	1-Day	LSTM	9.0
17	Mallqui and Fernandes (2019)	Bitcoin	1-Day	ANN	3.06
				RNN	3.36
18	Radityo et al. (2018)	Bitcoin	1-Day	BPNN	1.998
				GANN	4.461
				GABPNN	1.883
19	Jay et al. (2020)	Bitcoin	1-Day	MLP	3.06
				LSTM	3.20

5.7 Future research

These results point to some promising future research, first, the data-smoothing methodologies addition, for example, moving average filters in hybrid models, is entirely relevant and may be developed further by the addition of cryptocurrency prices and also by sentiment analysis of public perception. Second, establishment in future studies of which economic factors can explain significant variation in pricing is called for. Thirdly, the transfer learning area is a very young one within machine learning, and maybe in some time when, for example, new cryptocurrencies will be forecasted at their price, and the history information about them is short. Existing data of already established cryptocurrencies can already be used to help make better predictions for newer ones. There is also supposed to be much focus on the implementation of automation methodologies for the optimization of hyper-parameters, in order to find optimal sets of hyper-parameters. Other volatile factors which may affect Bitcoin Price and return can also be considered, which shall be prioritized in future studies.

6. Conclusion

This work is focused on an assessment of the applicability of LSTM neural networks in predicting Bitcoin close prices using historical price data. The dataset used for training and testing contained data related to daily prices of Bitcoin in USD from April 24, 2023, up to April 24, 2024. Preprocessing was done on the data by scaling the features using Min-Max normalization to ensure all the features were in one scale. It took the past 10 days of Bitcoin prices and additional features as input to predict the next day of Bitcoin closing prices. In addition, the dataset was split into a training and test set, where 90% is used for training and the remaining 10% for testing.

This LSTM model has been designed with an input rank-3 tensor: samples, 10-time steps, and 4 features. One added an LSTM layer with 2 hidden units, followed by a dropout layer to prevent overfitting at a rate of 20%. It also consisted of fully connected layers, where the last output layer would project the next day's price of Bitcoin. The model was trained for 1000 epochs using the Adam optimizer, setting the learning rate to 0.001, and the Mean Squared Error as the loss function in its optimization. Then, during training, the model was tracked on both the training and test losses, evaluated at intervals.

The mean absolute percentage error was then computed to clearly express the model predictive performance. From a simple comparison of the actual and predicted values, an LSTM model seems quite effective at capturing the trend of Bitcoin prices.

It reinforces the fact that good preprocessing of the data is enormously important, along with the choice of the structure of the model. How the hyper-parameter tuning involved the best performance of the model with the number of hidden units and learning rate further to know on dropouts. In essence, it contributed a measure that may validate the suitability of the Min-Max scaling method in the normalization of data to derive better convergence while training.

However, there were a couple of limitations during the process. First, in this study, only price data was considered for Bitcoin, without the inclusion of any other economic or financial variable that could give better context to the predictions. Second, the study assessed a relatively short period of time, which may further limit generality across different time frames. This did not include, for example, more advanced techniques that could be applied-sentiment analysis or transfer learning-to help in developing a better prediction accuracy.

Further research in predicting the future price of Bitcoin can be done by incorporating more economic indicators or sentiment analysis and coupling them with the model for better efficiency. The technique used can be Automated Hyperparameter Tuning. It works to find a combination of hyperparameters such that the most effective model configuration arises. This could be further investigated by considering even more complex and hybrid architectures using transfer learning to provide even better improvements in the accuracy of the predictions. This would be even more complete if it were to involve some volatility and return analyses with respect to the price dynamics of Bitcoin.

References

Altan, A. K. (2019). *. Digital currency forecasting with chaotic meta-heuristic.* Chaos, Solitons and Fractals. Retrieved from https://doi.org/10.1016/j.chaos.2019.07.011

Chiu, J. &. (2022). *The Economics of Digital Currencies: Progress and Open Questions.* Journal of Economic Dynamics and Control. Retrieved from https://www.sciencedirect.com/science/article/abs/pii/S0165188922002007?via%3Dihub

Chollet, F. (2021). *Deep learning with python* (Vol. Second). Manning Publications.

Coinmarketcap. (2022). Cryptocurrency Prices, Charts And Market Capitalizations. Retrieved from https://coinmarketcap.com/

De Oliveira Monteiro, A. H. (2019). *Market prediction in cryptocurrency: A systematic literature mapping.* PervasiveHealth: Pervasive Computing Technologies For Healthcare. Retrieved from https://dl.acm.org/doi/10.1145/3330204.3330272

Derbentsev, V. B. (2021). *Comparative Performance of Machine Learning Ensemble Algorithms for Forecasting Cryptocurrency Prices.* International journal of engineering. Retrieved from https://www.ije.ir/article_122162.html

Ferdiansyah, O. S. (2019). *A LSTM-Method for Bitcoin Price Prediction: A Case Study Yahoo Finance Stock Market.* ICECOS 2019 - 3rd International Conference on Electrical Engineering and Computer Science, Proceeding. Retrieved from https://ieeexplore.ieee.org/document/8984499/

Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow* (Second ed.). O'Reilly Media.

Giudici, G. M. (2020). *Cryptocurrencies: Market analysis and perspectives.* Journal of Industrial and Business Economics. Retrieved from https://link.springer.com/article/10.1007/s40812-019-00138-6

Gradojevic, N. K. (2021). *Forecasting Bitcoin with technical analysis: A not-so-random forest?* International Journal of Forecasting. Retrieved from https://www.sciencedirect.com/science/article/abs/pii/S0169207021001230?via%3Dihub

Graves, A. &. (2005). *Framewise phoneme classification with bidirectional LSTM and other neural network architectures.* International Joint Conference on Neural Networks 2005. Retrieved from https://www.sciencedirect.com/science/article/abs/pii/S0893608005001206?via%3Dihub

Hochreiter, S. &. (1997). *Long Short-term Memory.* Neural computation. Retrieved from https://direct.mit.edu/neco/article-abstract/9/8/1735/6109/Long-Short-Term-Memory?redirectedFrom=fulltext

Jay, P. K. (2020). *Stochastic Neural Networks for Cryptocurrency Price Prediction.* IEEE Access.

Klein, T. P. (2018). *Bitcoin is not the New Gold – A comparison of volatility, correlation, and portfolio performance.* International Review of Financial Analysis. Retrieved from https://www.sciencedirect.com/science/article/abs/pii/S105752191830187X?via%3Dihub

Lamothe-Fernandez, P. A.-L.-G. (2020). *Deep Learning Methods for Modeling Bitcoin Price.* Mathematics. Retrieved from https://www.mdpi.com/2227-7390/8/8/1245

Liu, M. L. (2021). *Forecasting the price of Bitcoin using deep learning.* Finance research letters. Retrieved from https://www.sciencedirect.com/science/article/abs/pii/S1544612320304864?via%3Dihub

Livieris, I. E. (2021). *An Advanced CNN-LSTM Model for Cryptocurrency Forecasting.* Electronics. Retrieved from https://www.mdpi.com/2079-9292/10/3/287

Mallqui, D. C. (2019). *Predicting the direction, maximum, minimum and closing prices of daily Bitcoin exchange rate using machine learning techniques.* Applied Soft Computing Journal. Retrieved from https://doi.org/10.1016/j.asoc.2018.11.038

Munim, Z. H. (2019). *Next-Day Bitcoin Price Forecast.* Journal of Risk and Financial Management. Retrieved from http://dx.doi.org/10.3390/jrfm12020103

Muriuki, G. (n.d.). *Medium*. Retrieved from https://towardsdatascience.com/deep-learning-personal-notes-part-1-lesson-2-8946fe970b95

Panagiotidis, T. S. (2018). *On the determinants of bitcoin returns: A LASSO approach.* Finance Research Letters. Retrieved from https://www.sciencedirect.com/science/article/abs/pii/S1544612318300023?via%3Dihub

Patel, M. M. (2020). *A Deep Learning-based Cryptocurrency Price Prediction Scheme for Financial Institutions.* Journal of Information Security and Applications. Retrieved from https://www.sciencedirect.com/science/article/abs/pii/S2214212620307535?via%3Dihub

Phaladisailoed, T. &. (2018). *Machine learning models comparison for bitcoin price prediction .* Proceedings of 2018 10th International Conference on Information Technology and Electrical Engineering: Smart Technology for

Better Society. Retrieved from https://ieeexplore.ieee.org/document/8534911

Polamuri, S. (n.d.). *LSTM: Introduction to Long Short Term Memory*. Retrieved from https://dataaspirant.com/lstm-long-short-term-memory/

Radityo, A. M. (2018). *Prediction of Bitcoin exchange rate to American dollar using artificial neural network methods.* 2017 International Conference on Advanced Computer Science and Information Systems. Retrieved from https://ieeexplore.ieee.org/document/8355070

Rizwan, M. N. (2019). *Bitcoin price prediction using Deep Learning Algorithm.* 2019 13th international conference on mathematics, actuarlal science, computer science and 2019 13th international conference on mathematics, actuarlal science, computer science and statistics.

Saad, M. C. (2020). *Toward characterizing blockchain based cryptocurrencies for highly accurate predictions.* IEEE Systems Journal. Retrieved from https://ieeexplore.ieee.org/document/8840919

Septiarini, T. W. (2020). *A comparative study for Bitcoin cryptocurrency forecasting in period 2017-2019. Journal of Physics: Conference Series.*

Tan, X. &. (2019). *Predicting the closing price of cryptocurrencies: A comparative study.* ACM International Conference Proceeding Series. Retrieved from https://doi.org/10.1145/3368691.3368728

Wang, Y. &. (2020). *Cryptocurrency price prediction based on multiple market sentiment.* Proceedings of the Annual Hawaii International Conference on System Sciences.

Zoumpekas, T. H. (2020). *ETH analysis and predictions utilizing deep learning.* Expert Systems with Applications.