NOVA IMS Information Management School

MGI

Master Degree Program in
Information Management

Bitcoin Price Prediction

A Comparative analysis of PatchTST and LSTM

Thomás Martino Martins Dutra

Master Thesis

presented as partial requirement for obtaining the Master Degree in Information Management

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

Bitcoin Price Prediction

A Comparative analysis of PatchTST and LSTM

by

Thomás Martino Martins Dutra

Master Thesis presented as partial requirement for obtaining the Master's degree in Information Management, with a specialization in Business Intelligence.

Supervised by

Professor Ian James Scott

July, 2024

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration. I further declare that I have fully acknowledged the Rules of Conduct and Code of Honor from the NOVA Information Management School.

Lisbon, 03/07/2024

DEDICATION

I would like to express my gratitude to everyone who has supported me throughout this academic journey.

First and foremost, I am really greatfull for my family, who have supported with their love and encouragement. You have always had faith in me, even when I doubted myself, and that has made all the difference. I am forever grateful for the sacrifices you made to provide me with this opportunity.

To my fiancé, Bárbara, thank you for being my rock through all the highs and lows of this journey. Your patience, understanding, and unwavering support were vital in helping me complete this dissertation. I am grateful for your constant encouragement, for listening to me, and for believing in me even when I struggled. Your presence has made this journey more manageable and meaningful.

Lastly, but certainly not least, I extend my gratitude to my advisors, professors, and research colleagues. Your knowledge and guidance have been instrumental in the completion of this work. Without your support and dedication, this dissertation would not have been possible.

To all of you, my most sincere thanks.

ABSTRACT

Transformers-based algorithms have revolutionized the field of natural language processing. In this context, these types of algorithms are being adapted to time series forecasting. The present study compares PatchTST, a Transformers-based algorithm, against the commonly used Long Short-Term Memory (LSTM) algorithm in the problem of Bitcoin forecasting. The methodology employed in this study involved the collection of data from blockchain, macroeconomics, and technical indicators. This data was used to predict Bitcoin prices across various lookback and horizon windows. The models were evaluated using standard evaluation metrics to ensure robust performance comparison. The results demonstrated that PatchTST outperformed LSTM in both short and long-term horizons. Specifically, for the short-term horizon, PatchTST achieved a Root Mean Square Error (RMSE) of 10.36 compared to 149.72 for LSTM. For long-term horizons, PatchTST achieved an RMSE of 1 909.96, whereas LSTM had an RMSE of 2 310.41. PatchTST exhibits superior capability in capturing and predicting Bitcoin prices compared to LSTM, making it a more effective algorithm for time series forecasting in the context of cryptocurrency markets.

KEYWORDS

Bitcoin; Cryptocurrencies; Machine Learning; Long Short-Term Memory, Transformers, PatchTST;

Sustainable Development Goals (SDG):

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

BTC	Bitcoin
LSTM	Long Short-Term Memory
ARIMA	Autoregressive Integrated Moving Average
AR	Autoregressive
MA	Moving Average
MSE	Mean Square Error
MAPE	Mean Absolute Percentage Error
ANN	Artificial Neural Network
RNN	Recurrent Neural Network
OHLC	Open, High, Low and Closed
GRU	Gated Recurring Unit
SVM	Support Vector Machine
KNN	K-Nearest Neighbors
TFT	Temporal Fusion Transformer
MACD	Average Convergence Divergence
RSI	Relative Strength Index
ADX	Average Directional Index
SMA	Simple Moving Average
VIX	Chicago Board Options Exchange Volatility Index
EMA	Exponential Moving Average
OBV	On Balance Volume
MAE	Mean Average Error
RMSE	Root Mean Squared Error
MAPE	Mean Absolute Percentage Error

1. INTRODUCTION

Since its inception, Bitcoin (BTC) has drawn the attention of investors, traders, and researchers for being a financial asset and a digital currency with extreme volatility and rapid price fluctuations (J. Chen, 2023). Bitcoin operates on a decentralized peer-to-peer network on blockchain, and unlike any traditional fiat currency, it is neither regulated nor issued by any central authority (Nakamoto, 2008).

Price fluctuations in digital currencies naturally cause tension between investors, academics, and regulators. The anonymous, decentralized, and unregulated cryptocurrency markets may endanger financial stability (Atsalakis et al., 2019). Thus, there is a need for accurate and reliable price forecasting.

The first studies conducted on Bitcoin have focused on applying a statistical approach to understand the factors influencing BTC price formation (Cheung et al., 2015). More recently, studies have applied machine learning models to the task (J. Chen, 2023; Hua, 2020; Shin et al., 2021). In this context, different models have been adapted to time series forecasting, making them possible to apply to Bitcoin price prediction (Goodell et al., 2023).

Transformers-based models like GPT and Bart have redefined the boundaries of natural language processing (Khan et al., 2022). In this context, Transformers models have been adapted to time series forecasting making them possible to apply to Bitcoin price prediction (Ramos Pérez et al., 2021; Wen et al., 2023). However, there still exist some questions about the effectiveness of Transformers time series forecasting (Zeng et al., 2023).

In response, the PatchTST algorithm, which is a Transformer base model (Nie et al., 2023), was introduced as an effective method for time series prediction using a Transformer, especially when a long horizon is forecasted. The authors have benchmarked PatchTST against other Transformers-based algorithms and have achieved reliable results on a variety of datasets.

In this context, PatchTST remains relatively unexplored in the context of Bitcoin price prediction. To address those needs, the present study aims to answer: "Can the PatchTST algorithm, based on the Transformer architecture, provide accurate predictions of Bitcoin's price movements when compared to the commonly employed Long Short-term Memory (LSTM) algorithm?". To correctly answer this question, the main objectives of this research are: 1) Assess the accuracy of the PatchTST algorithm in predicting Bitcoin price fluctuations using technical, macroeconomic, and blockchain indicators. 2) Conduct a comparative analysis by benchmarking the PatchTST algorithm against the LSTM algorithm.

A comparative approach is applied using historical Bitcoin price data, as well as technical, macroeconomic, and blockchain indicators. The models are then trained and evaluated, comparing their performance.

On long sequences like Bitcoin price forecasting, LSTM models can suffer from vanish gradient and then failing to capture dependencies in the data (Hochreiter, 1998), so PatchTST is expected to perform better under these circumstances. The contribution to researchers and traders is the comparison of performance between PatchTST and LSTM on a real-world challenge, enhancing the applicability of machine learning on financial markets.

2. LITERATURE REVIEW

This section aims to present an overview of the work done in Bitcoin price forecasting. The goal is to explain the methods that previous works have employed as well as the conclusions that can be drawn from them.

Prediction and analysis of financial markets involve the use of time series data (Montgomery et al., 2015). The authors explain that a time series is a chronological sequence of observations. The task of time series prediction can fall into two categories: statistical and machine learning methods (Shah et al., 2019), hence, a distinction will be made between the work done between statistical methods and machine learning approaches. (Makridakis et al., 2018) affirm that statistical and machine learning approaches have the same objective, minimize the sum of errors. The difference is that while the statistical method uses linear process, machine learning uses non-linear algorithms.

Lastly, in this section, a comparative review is done on the features employed in the task of BTC forecasting. The goal is to provide insights into what variables are most useful for the task of this work.

2.1. STATISTICAL METHODS

The traditional approach for time series forecasting is the statistical method, where statistical formulas and theories are applied to capture patterns in the time series (Murray et al., 2023). This technique often assumes stationarity (Shah et al., 2019). A time series process is considered to be strictly stationary if it has constant mean, variance and covariance over time, however if the mean and covariance are time shift invariant is considered to be weakly stationary process (Gujarati & Porter, 2009). However (Rao & Gabr, 1984) affirm that the majority of time series problems in the real world do not follow stationarity process, hence, further adjustment is needed to apply this type of technique to Bitcoin forecasting.

The Autoregressive Integrated Moving Average (ARIMA) model adapts a stationarity time series into a non-stationary one by accounting for trends in the data (Box et al., 2015). ARIMA originated from Autoregressive moving Average (ARMA) where it combines Autoregressive (AR) and Moving Average (MA) models (Shah et al., 2019). The integrated process and differencing it one or more times are responsible for turning ARMA into a non-stationary time series (Gujarati & Porter, 2009).

Initial research primarily employed statistical and econometric techniques aimed at understanding the variables driving Bitcoin's price and fundamental characteristics (Ji et al., 2018). Utilizing ARMA (Szetela et al., 2016) aimed to understand the dependency between Bitcoin and selected global currencies. The researchers concluded that bitcoin shows no relationship among the dependent variables studied.

For short-term prediction of Bitcoin returns, the work of (Munim et al., 2019) and (Wirawan et al., 2019) concluded that ARIMA produces reliable predictions, however, as the predicted horizon increases, the Mean Square Error increases (MSE), which decreases the reliability of the forecasting model.

Adding to the study (Latif et al., 2023), compared how effective are the predictions of ARIMA and Long Short-Term Memory (LSTM) on short-term Bitcoin price forecasting. The conclusions are that ARIMA could only track the trend of Bitcoin prices, while LSTM forecasts were much closer than historical prices, the Mean Absolute Percentage Error (MAPE) of ARIMA was 1,79% against 0,27% of LSTM. This conclusion agrees with (Hua, 2020) who concluded that despite the downside of a longer training process, LSTM predictions are better than ARIMA for short-term time frames.

Using Boruta and forward and backward stepwise feature selection (Z. Chen et al., 2020) benchmarked machine learning and statistical approaches to forecast bitcoin prices. Their findings pointed out that Logistic Regression does a better job of predicting BTC prices on daily time intervals. However, for a five-minute time frame, Long Short-Term Memory has the best results among the algorithms employed with accuracy of 67% on price direction using LSTM while Logistic Regression has a 54,5% of accuracy.

Other researches have concluded that machine learning outperforms ARIMA in terms of prediction accuracy, (McNally et al., 2018) have compared the accuracy of LSTM and ARIMA on predicting the price direction of Bitcoin. The results pointed to LSTM having the highest classification accuracy of 52,78% and a RMSE of 6,87% while ARIMA 50,05% and 53,74% respectively. This conclusion agrees with (Shin et al., 2021) which compared the accuracy of a ensemble model of minute, hour and daily time frames using LSTM and ARIMA. This is related to the fact that ARIMA is designed to handle linear data relationships, thus fails to deal with complex non-linear time series (Kontopoulou et al., 2023) which is the case of Bitcoin. The nature of time series price data requires the use of more sophisticated methods to capture the complex nonlinear relationships between data variables (Shin et al., 2021). It is important to mention that this work will not use ARIMA as a target model, instead, as aforementioned work, it is demonstrated that although ARIMA is commonly employed in time series forecasting, it does not provide better results than LSTM.

2.2. MACHINE LEARNING METHODS

Machine learning is a subset of the field of artificial intelligence (Kontopoulou et al., 2023). Machine learning allows complex models to learn data relationships by examples, this knowledge allows algorithms to accomplish certain tasks (Tiwari et al., 2021).

In machine learning problems, the tasks are commonly classified in supervised and unsupervised learning, where the first receives a set of labeled input data to train the algorithm and the second receives only unlabeled data (Shah et al., 2019). Financial time series forecasting is included in supervised learning where previous data points are fed into the model to produce the forecasted results. In ML The objective is to develop algorithms that automatically improve by practice, where the more the model is run, the more accurate the outcome becomes (Staudemeyer & Morris, 2019).

Inspired by biological neuron structure, (Jabbar et al., 2021) explains that the Artificial Neural Network (ANN) is a class of data processing algorithms composed by artificial neurons. The neurons or nodes, work to process data, and the nodes work as information carriers, additionally, the flow of information between nodes is generally referred to as activation function. The model is represented by a system of artificial neurons and nodes, moreover, ANNs are commonly arranged in form the of layers, having the output of one layer serving as the input to the next layer and possibly other layers, the layers are linked by weighted connections. The input layer receives input data and feeds it into the network. The input layer processes information that is then transmitted to one or more hidden layers, where the actual data processing takes place. The output layer receives the information from the hidden layer (Jabbar et al., 2021).

However, ANN does not provide a way to account for time-space relationship in the architecture. A machine learning class algorithm that can model information over time steps is the Recurrent Neural Network (RNN). RNNs are trained to generate output where the predictions at each time step are based on current input and information from the previous time steps (Singh et al., 2022).

The most used neural network architecture for sequence prediction is the RNNs (Hewamalage et al., 2021). This class of algorithm has gained attention in the domain of natural language processing, one factor that distinguishes them from ANNs is that RNNs have feedback loops of the recurrent cells that address the temporal order and the temporal dependencies of sequences, unlikely ANNs, Recurrent Neural Network process information in two way flows, information flows forward in time, but the hidden state is updated recurrently, allowing past information to influence future points (Schäfer & Zimmermann, 2006).

A major problem as (Hochreiter, 1998) explains is the issue related to the use of RNNs in long sequences. In the process of learning which past inputs must be stored, Recurrent Neural Networks must calculate the current error signal that has to “flow back in time” over the feedback connections, however, due to the insufficient weights changes the long-term dependencies are not learned appropriately, making the gradient to vanish.

Long Short-Term Memory (LSTM) algorithm emerged as a solution to effectively learn data dependencies by addressing the vanishing gradients problem (Hochreiter & Schmidhuber, 1997). The authors introduced the use of two components that can address the problem mentioned. LSTM uses memory cells and gate units in its architecture. Memory cells allow for constant error flow and hold information for an extended period and gate units control the flow of information. Figure 1 is the graphical representation of LSTM unit.

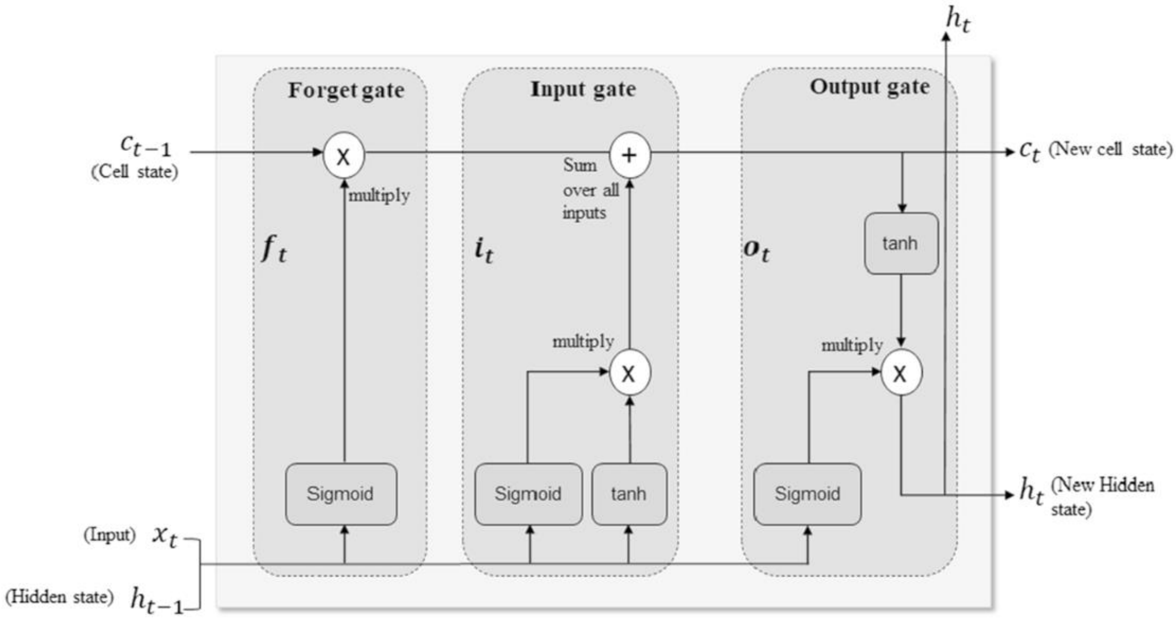

Figure 1 – LSTM unit structure, illustrating the forget gate (f_t), input gate (i_t), and output gate (o_t) along with their interactions with the cell state (c_t) and hidden state (h_t). (Tripathi & Sharma, 2022).

Figure 1 presents the core idea of LSTM unit cell, which is represented by the two black horizontal lines, there is where information flows. The bottom one is where short-term information flows and is the combination of input of the current time x_t and the output of the previous hidden state time h_{t-1} (Graves & Schmidhuber, 2005). The top line is where the long-term information flows, and it is adjusted during the process. As mentioned, gates in LSTM control the flow of information. The forget gate(f_t) takes the input of previous hidden state (h_{t-1}) and passes it through an activation function ($Sigmoid$) combined with the input of the cell state (c_{t-1}) to create the forget gate output. The activation function regulates how much information of short-term will be passed to long-term (Graves & Schmidhuber, 2005). The input gate (i_t) updates the cell state by combining the previous hidden state (h_{t-1}) and the current state (x_t) by a series of non-linear functions ($Sigmoid\ and\ Tanh$), in this process, it is determined how should be updated the cell state (c_t). The output gate (o_t) takes the input gate (i_t) information and passes through a set of non-linear functions to determine what information will be passed forward.

LSTM has produced reliable results and has been benchmarked against multiple models on Bitcoin forecasting in the past literature (Shin et al., 2021).

Studying various machine learning models, (Aggarwal et al., 2019) predict bitcoin price fluctuation using Root Squared Mean Error (RMSE) as a parameter to determine which algorithm would produce better results. The results pointed out that LSTM had the least RMSE, being the best model studied. The RMSE of LSTM was 151,67 and the Gated Recurrent Unit was 179,23.

Comparing LSTM to ARIMA and RNN, (McNally et al., 2018) study whether these algorithms could predict the price direction of BTC. The study concluded that both machine learning algorithms produce better results than ARIMA, additionally, LSTM presented a higher accuracy than RNN. Moreover, the authors concluded that the training time can be reduced by 67% if the algorithm is trained on GPU instead of CPU.

The accuracy of the LSTM model prediction can be increased by 14.7% if 10-fold cross-validation is done (Tandon et al., 2019), however the authors used only Open, High, Low and Closed (OHLC) bitcoin data, which can limit the performance on other models.

Enlarging the study on this field, (Cocco et al., 2021) created a machine learning framework consisting of two stages. The first stage was set to predict the variables to a certain time in the future and the second stage to predict the bitcoin price. The algorithms used in the experiment were the Bayesian Neural Network, Feed Forward Neural Network, and Long Short-Term Memory. The results pointed out LSTM being the best performing on two stages framework.

In one hand, the study of (Dutta et al., 2020) focused on comparing the results of Gated Recurring Unit (GRU), Neural Network, and LSTM. The authors concluded that GRU outperforms the before-mentioned algorithms. Contrastly, (Pichaiyuth et al., 2023) studied if Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Random Forest, and Naïve Bayes have a better performance on price trend forecasting. The results pointed out that LSTM exhibits the most optimal performance for 90 days horizon.

Recent studies have applied different approaches in Bitcoin price forecasting. (J. Chen, 2023) benchmarked the accuracy of predictions between LSTM and Random Forest Regression. The author used a wide range of features as inputs, including macroeconomic, technical, sentiment and blockchain information. The results show that Random Forest performed better than LSTM with RMSE of 3,48% against 5,05% RMSE of LSTM model on the latest period studied. However, the author highlights the major drawback of Random Forecast Regression is the inability to predict the results that did not appear in the training samples, meaning, that the model will not be able to predict new price highs.

Using a different approach, the study of (Goodell et al., 2023) study a similar set of features on Bitcoin price forecasting. The authors applied a new feature selection method integrating the game-theory-based SHapley Additive exPlanations to assess how each attribute contributes to the final model prediction. The variables that are most relevant for predicting BTC prices are Google Trends for Recession, Geopolitical Risk, Five Year Forward Inflation Expect, and S&P 500. Among the algorithms used, LSTM produced the most reliable results with a RMSE of 2,031% using the feature selection proposed.

A new architecture for dealing with machine translation, language modeling, and sequence modeling brought by (Vaswani et al., 2017) has revolutionized the field of natural language processing and computer vision (Lin et al., 2022) the most popular ones include Bidirectional

Encoder Representations from Transformers (BERT) and Generative Pre-trained Transformer (GPT) (Khan et al., 2022).

The Transformer architecture contains an encoder-decoder structure and utilizes attention mechanism to predict future data points. Utilizing positional encodings, the input sequence captures the position of each data point, the relative position can be defined by sine or cosine functions. The decoder, however, uses the inputs from the encoder and previous outputs to predict the next data point in the sequence. The attention mechanism estimates the relevance of one data point to other data points (Vaswani et al., 2017).

Transformers have been proven to achieve great results for long-range dependencies and interactions in sequential data, thus appealing for time series modeling (Wen et al., 2023). Variants of Transformer architecture haven been proposed to address the time series forecasting problems (Wen et al., 2023).

On financial markets, the work of (Ramos Pérez et al., 2021) proposes a volatility forecasting method using Transformers, the results pointed out that combining two vanilla Transformers and Multi-Transformers achieves the best results among the methods utilized.

Another attention-based algorithm has been used on financial time series forecasting. The work of (Lim et al., 2021) presents the Temporal Fusion Transformer (TFT) algorithm and its applicability in a wide range of datasets. The authors implement quantile regression to obtain output intervals across all prediction horizons. To evaluate results, the quantile losses were computed and TFT model is benchmarked against ARIMA in different contexts, including financial stock volatility, where TFT outperformed ARIMA. The loss of ARIMA was 0.0154 while TFT was 0.055 for horizon of 50 timesteps. For the 90 timesteps horizon the loss was 0.102 and 0.020 respectively.

Contrastingly, in cryptocurrency forecasting, the subject of this study, (Murray et al., 2023) and (Lopes & Bianchi, 2022) show that the TFT algorithm did not have the best performance. (Murray et al., 2023) shows that LSTM has a better performance in predicting the price of Bitcoin when compared to TFT, ARIMA, SVR, GRU, TCN, and ARIMA. Adding to the problem, the work of (Lopes & Bianchi, 2022) compared the results of the second most known cryptocurrency, Ethereum. The results pointed to ARIMA to be better suited to Ethereum price forecasting.

However, there is still some debate on whether Transformers perform well on time series forecasting. While the work of (Zeng et al., 2023) argues that a simple linear model can outperform other Transformers-based architecture models in a variety of common benchmarks, the work of (Nie et al., 2023) presents a new approach to time series forecasting based on Transformers architecture, which outperforms the linear model and others Transformers-based algorithms.

PatchTST is introduced by (Nie et al., 2023). It is based on two main components: patching the time series into subseries which are served as input tokens to Transformers and channel-independence where each channel contains a single univariate time series.

The authors explain that by patching the time series into subseries the model would capture the correlation between data in each different time step, instead of processing a single time step at a time, making possible to remember long term correlations, thus enhancing the performance on long forecast horizons. The authors use the example of a word that does not provide a local semantic in a sentence, so is needed more information to understand the relationship among data points. In contrast to vanilla Transformer, PatchTST does not use point-wise aggregation, instead, the model enhances the locality and captures comprehensive semantic information that is not available in point-level by aggregating time steps into subseries-level patches. Additionally, by patching the model memory usage is reduced maintaining the same look-back window.

The second key point of the algorithm is channel-independence. In the model created by (Nie et al., 2023) a multivariate time series is treated as a multi-channel signal, and each input token contains information from a single channel.

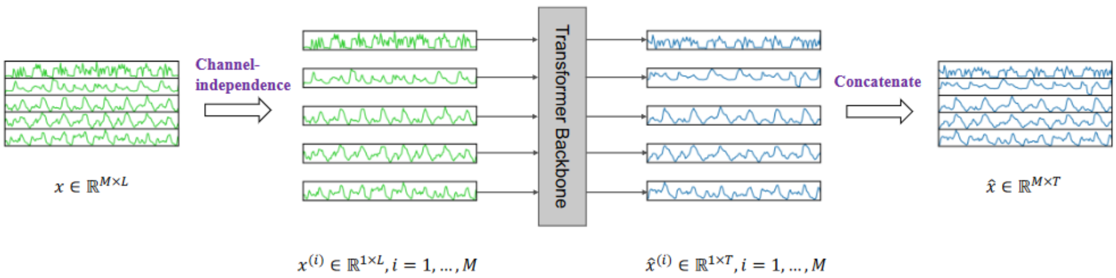

Figure 2 - PatchTST model overview. A multivariate time series is divided into univariate time series, where each time series corresponds to one channel. Each univariate series is independently processed through the same Transformer backbone. Each univariate time series is forecasted, to be further concatenated to generate the multivariate time series output (Nie et al., 2023).

The forward process of passing a multivariate time series into univariate time series, PatchtTST model starts with a multivariate time series data $x \in \mathbb{R}^{MxL}$, where M represents the number of channels and L is the length of the lookback window. Each time step $x_t \in \mathbb{R}^M$ in this window is a vector of dimension M. The multivariate series is split into M univariate time series $x^{(t)} \in \mathbb{R}^{1\times L}$, with each univariate series corresponding to one channel. The univariate series are fed into the Transformer backbone to produce predictions $\hat{x}^{(i)} \in \mathbb{R}^{1\times T}$ for each time series, where T is the forecasting horizon. Finally, the predictions are concatenated to create the overall multivariate prediction $\hat{X} \in \mathbb{R}^{M\times T}$. $\hat{X}$ is the matrix representing the forecasted future values for all M channels over T time steps (Nie et al., 2023).

In the Transformer backbone a series of transformations happen to the univariate input timeseries. The transformations are instance normalization, patching, positional embedding, transformer encoder, and output processing. Figure 3 exhibits the transformer backbone architecture.

The inputs, univariate timeseries of $x^{(t)} \in \mathbb{R}^{1\times L}$, are normalized to have zero mean and unit standard deviation. After, each normalized series $x^{(i)}$ are divided into patches that can be either overlapped or non-overlapped. Denote P as patch length and the stride – the non-overlapping region between two consecutive patches as S. Then the patching process will generate a sequence of patches $x_p^{(i)} \in \mathbb{R}^{P\times N}$ where N is the number of patches of $N = \left\lfloor \frac{(L-P)}{S} \right\rfloor + 2$. The authors affirm that the use of patching allows the model to look to a longer historical sequence, which can result in an improvement of prediction performance.

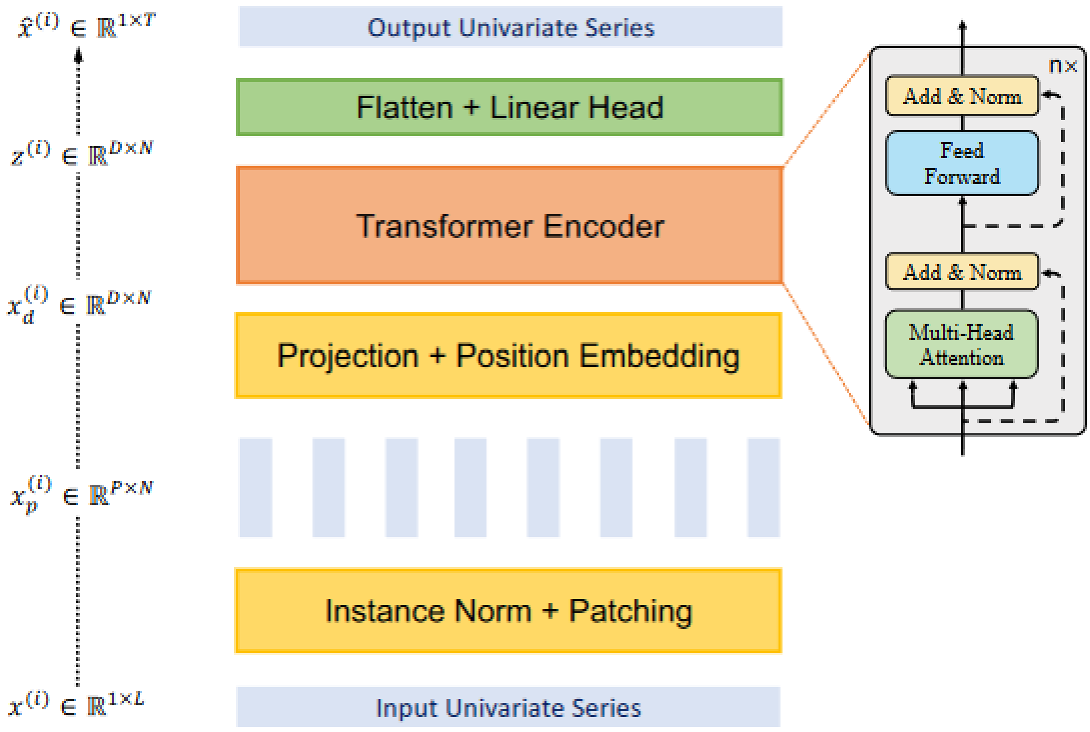

Figure 3 - Transformer Backbone (Nie et al., 2023)

After the patching process, the model uses vanilla Transformer encoder that maps the observed signals to the latent presentations. The patches are mapped to the Transformer latent space of dimension D via a trainable linear projection $w_p \in \mathbb{R}^{D\times P}$ and a learnable additive position encoding $w_{pos} \in \mathbb{R}^{D\times N}$ is used to guarantee the temporal order of patches: $x_d^{(i)} = W_p x_p^{(i)} + W_{pos}$, where $x_d^{(i)} \in \mathbb{R}^{D\times N}$ is the input that will be fed into Transformer encoder.

Each head (h) in the multi-head attention block, inside the Transformer encoder, transform inputs into query matrices $Q_h^{(i)} = \left(x_d^{(i)}\right)^T W_h^Q$, key matrices $K_h^{(i)} = \left(x_d^{(i)}\right)^T W_h^K$, and value matrices $V_h^{(i)} = \left(x_d^{(i)}\right)^T W_h^V$, where $W_h^Q, W_h^K \in \mathbb{R}^{D \times d_k}$ and $W_h^V \in \mathbb{R}^{D \times D}$. After that a scaled production is used for getting attention output $O_h^{(i)} \in \mathbb{R}^{D \times N}$: $\left(O_h^{(i)}\right)^T = Attention\ (Q_h^{(i)}, K_h^{(i)}, V_h^{(i)}) = Softmax(\frac{Q_h^{(i)} K_h^{(i)T}}{\sqrt{d_k}})V_h^{(i)}$.

Batch norm layers and a feed forward network with residual connection are included in the multi-head attention block as shown in figure 3. Afte, it generates the presentation denoted as $z^{(i)} \in \mathbb{R}^{D \times N}$. Finally, a flatten layer with linear head is used to obtain the prediction result $\hat{x}^{(i)} = (\hat{x}_{L+1}^{(i)}, \ldots, \hat{x}_{L+T}^{(i)}) \in \mathbb{R}^{1 \times T}$.

The model is benchmarked against state of art Transformers algorithms to check its viability. The model has achieved the least Mean Squared Error in six popular datasets that are commonly used in benchmarking, in the remaining two datasets, the new algorithm had the second-best performance.

2.3. FEATURES EMPLOYED IN BITCOIN FORECASTING

In the academic field there is not a well-established set of features to predict Bitcoin price behavior as in studies, authors use different sets of data to accomplish the task of BTC forecasting. As described by (Tripathi & Sharma, 2022), previous studies can be categorized in studies that employ only historical prices as inputs, or studies that use a high number of features as data predictor factors.

In the first category, the study of (Seabe et al., 2023) uses High, Open, Low, Close, and Volume (OHLCV) to predict bitcoin and other cryptocurrencies. Additionally, (Parvez et al., 2022; Tiwari et al., 2021) use OHLCV and the volume in Bitcoin traded, which can fail to capture the necessary attributes to properly forecast BTC price formation.

Adding to the problem, (Pichaiyuth et al., 2023) use a set of technical indicators, which are calculated using OHLCV data, the technical indicators used are: Average Convergence Divergence (MACD), Relative Strength Index (RSI), Stochastic RSI, Williams Percent Range, Price Rate of Change, Commodity Channel Index, Average Directional Index (ADX), and Simple Moving Average (SMA). (Huang et al., 2019) further explore the power of technical indicators as predictors by creating a decision-tree based model using 124 technical. The findings pointed out that technical indicators have a strong predictability power on BTC price movements.

Using network information (Z. Chen et al., 2020) forecast Bitcoin. The information on the BTC network used was Average Block Size in MB, Hash Rate, Mining Difficulty, and time between mined blocks in minutes. The work of (Aalborg et al., 2019) also uses network information to understand BTC behavior, concluding the changes in the number of unique addresses used on the Bitcoin network is positively related to Bitcoin returns. (W. Chen et al., 2021) goes further and add mining profitability and market capitalization as feature forecasters.

Using social sentiment data, (Serafini et al., 2020) investigate the predictive power of network sentiments. The authors use tweet sentiment data as well as the volume of tweets on BTC. The authors found that by including social data increases the prediction accuracy on models. (J. Chen, 2023) also uses Twitter data and includes Google Trend search into the analysis. (Goodell et al., 2023) further explores the use of Google Trends by adding search volume of the terms “Inflation” and “Recession”.

In macroeconomic analysis, (Yen & Cheng, 2021) study whether economic policy uncertainty influences cryptocurrency volatility, the authors found that policy uncertainty predicts BTC volatility. Further enhancing the study on the problem, (Nagula & Alexakis, 2022), use the closing price of the S&P500 and Treasury bond rates to create a model for predicting BTC price, showing that the S&P500 is highly correlated with Bitcoin volatility.

Using gold and oil prices, (Basher & Sadorsky, 2022) study the volatility of BTC the findings point out that gold and oil are important predictors to understand BTC fluctuation. (W. Chen et al., 2021) also use macroeconomic features as Chicago Board Options Exchange Volatility Index (VIX) and NASDAQ on their study.

Table 1 compares the features employed on the task of Bitcoin forecasting.

Table 1 – Literature Review on Features for BTC Forecasting.

Type	Indicator	Authors
Technical Information	MACD RSI StochRSI ROC SMA Bollinger Bands EMA	(Pichaiyuth et al., 2023), (Huang et al., 2019), (Tripathi & Sharma, 2022), (Wang et al., 2023)
Network Information	Block size Hash rate Mining difficulty Time between blocks Number of transactions Total transaction fees Unique addresses	(Z. Chen et al., 2020), (Aalborg et al., 2019), (W. Chen et al., 2021), (J. Chen, 2023), (Nagula & Alexakis, 2022), (Tripathi & Sharma, 2022), (Erfanian et al., 2022)
Social Data	Tweets Sentiment Tweets Volume Google Trends for Bitcoin Google Trends for Inflation Google Trends for Recession	(Serafini et al., 2020), (J. Chen, 2023), (Goodell et al., 2023), (Tripathi & Sharma, 2022), (Aggarwal et al., 2019)

Macroeconomic	S&P500 Index VIX EPU DOW30 Index NASDAQ Index Gold Price Oil Price Treasury Bonds Bitcoin Futures Close Price Inflation Initial Clams US	(Yen & Cheng, 2021), (Nagula & Alexakis, 2022), (Basher & Sadorsky, 2022), (W. Chen et al., 2021), (J. Chen, 2023), (Goodell et al., 2023), (Tripathi & Sharma, 2022), (Erfanian et al., 2022), (Wang et al., 2023)

It can be concluded that there is not a well-defined set of features to predict BTC prices. The choice of features is typically based on the empirical knowledge of the researcher and its applicability to the forecasting model.

2.4. RESEARCH GAP

The recent success of language models like ChatGPT and BARD has raised questions if the same type of outcomes could be achieved when applied to time series forecasting. Recent studies have adapted the same architecture of language models to time series prediction, however, there is still uncertainty about the reliability of results when compared to traditional approaches (Zeng et al., 2023).

Cryptocurrencies like Bitcoin have increasingly become part of financial markets over the last few years. Researchers, politicians, and traders seek to understand BTC market dynamics to make informed decisions. Most studies on cryptocurrency forecasts have used traditional approaches to achieve this task and ignore the potential predictability power of Transformers in capturing time-dependent patterns.

The present study aims to fill the gap by introducing the PatchTST on Bitcoin forecasting. The existing literature has commonly applied traditional machine learning models for cryptocurrency price fluctuations. However, few studies have explored the capabilities of Transformers-based algorithms on bitcoin forecasting. The gap exists in the exploration of how PatchTST performs in comparison to LSTM in the Bitcoin forecasting task.

The limitations of LSTM model, which can suffer from vanishing gradient on sequences longer than 1000 steps (Hochreiter, 1998), highlight the need for alternative approaches. Transformers-based algorithms in this essence offer a potential solution for this challenge of capturing long-term dependencies.

3. METHODOLOGY

In the methodology section of this master thesis, the approach taken to benchmark LSTM and PatchTST algorithms for Bitcoin price forecasting is meticulously outlined. Figure 4 represents the diagram of the methodology taken in this work.

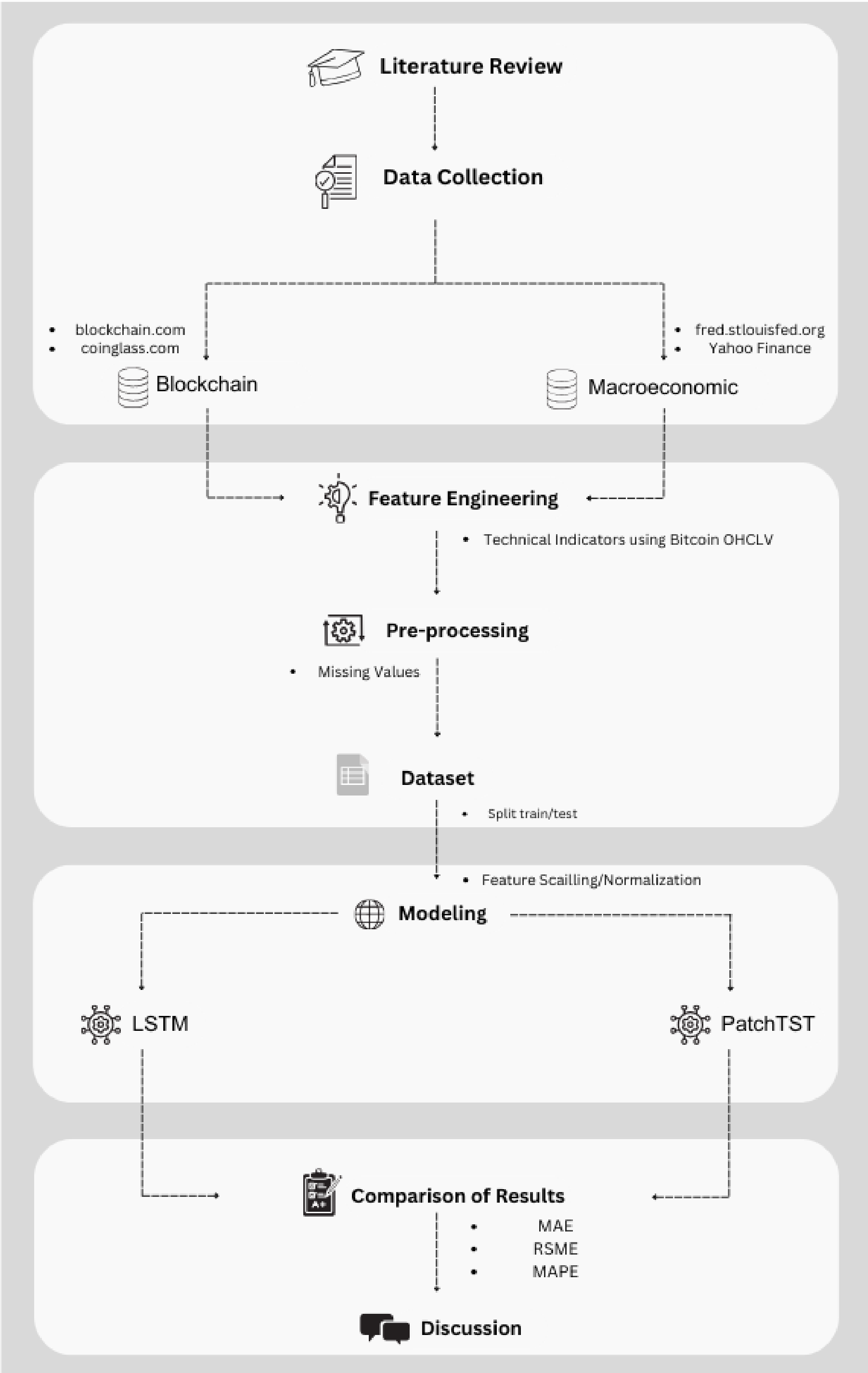

Figure 4 - Study Organogram

This process involves several key stages. Starting with an extensive literature review already done in chapter 2 to understand the current state of research in this area and identify relevant explanatory variables. Next, the data collection phase is done by gathering data from multiple sources. Using OHCLV Bitcoin price a set of technical indicators is built on feature engineering stage, followed by preprocessing task to input missing values. Once the dataset is created, the data is split into train and test sets. The models are then trained, and results are compared.

3.1. Data Collection

On the literature review is possible to identify three groups of variables that are used in the task of Bitcoin forecast. Blockchain, macroeconomics and technical information. The technical data will be calculated after the data is collected and it will be further explained. The data collection is done by either downloading the date from a website or by accessing it through a python package. Table 2 lists the public available data that is used in the study as well as source.

Table 2 – Data collection sources

Feature Type	Name	Source
Blockchain	Average Block Size (MB)	https://www.blockchain.com/
	BTC Dominance	https://bitcoinvisuals.com/
	Confirmed Transactions Per Day	https://www.blockchain.com/
	Market Value to Realized Value (MVRV)	https://www.blockchain.com/
	Miners Revenue (USD)	https://www.blockchain.com/
	Network Difficulty	https://www.blockchain.com/
	Network Value to Transactions Signal (NVTS)	https://www.blockchain.com/
	Total Hash Rate (TH/s)	https://www.blockchain.com/
	Unique Addresses Used	https://www.blockchain.com/
Macro	Bitcoin historical OHLCV	Yahoo Finance python library
	Chicago Board Options Exchange Volatility Index (VIX)	Yahoo Finance python library
	Gold	Yahoo Finance python library
	Oil	Yahoo Finance python library
	S&P 500 Closed Price	Yahoo Finance python library
	US Federal Funds Rate	https://fred.stlouisfed.org/
	US Initial Jobless Claim	https://fred.stlouisfed.org/
	US One Year Expected Inflation Rates	https://fred.stlouisfed.org/
	US Treasury Yield 10 Years	Yahoo Finance python library

The period under analysis is from 04/04/2015 until 14/03/2024. The features chosen are based on the literature review previously done. The Open, High, Low, Closed and Volume Bitcoin data will be used to calculate the technical indicators.

3.2. Feature Engineering

Once the data is collected, a set of technical indicators can be calculated using the OHLCV Bitcoin data. As seen in the literature review, technical indicators have been widely used in the task of BTC forecasting. The technical indicators are calculated using the TA-lib library that is broadly used for financial analysis. The features calculated are:

- Exponential Moving Average (EMA) for 20, 50 and 200 periods
- Average Directional Index (ADX)
- Bollinger Bands
- Moving Average Convergence Divergence (MACD)
- On Balance Volume (OBV)
- Relative Strength Index (RSI)
- Stochastic Oscillator

3.3. Pre-Processing

In this study the variables under analysis belong to cryptocurrencies market as well as stock market. Cryptocurrencies can be traded 24 hours per day while trading stocks, commodities and other securities are bound to stock market exchange providers (Dutta et al., 2020). Thus, this leads to a lack of information during weekends. To avoid missing values the feed forward imputation is adopted, whereas the last point value before a missing value is passed forward.

3.4. Dataset

The final dataset is composed of 3268 rows and 36 variables, each row represents a daily time stamp. Before applying the forecast algorithm, the dataset is divided into two intervals. The same approach was taken in the study of (Nagula & Alexakis, 2022), (J. Chen, 2023), and (Munim et al., 2019), where Bitcoin forecast was divided into two periods, in the study of (Tripathi & Sharma, 2022) e (Erfanian et al., 2022) the BTC dataset was divided into four intervals.

After the dataset is divided into two different periods that is possible to split into train and tests sets. As time series are a chronological sequence of observations (Montgomery et al., 2015) the split step should obey the chronological order of the data point, in this way, it will avoid leaking information.

The periods are divided to have the same length of data points. The first period will range from 04/04/2015 to 26/08/2019 and the second period from 21/10/2019 to 14/03/2024. Figure 5 represents how the data is divided.

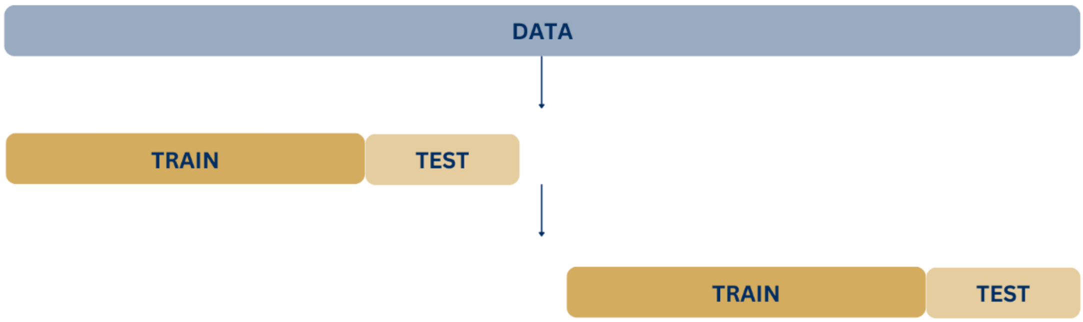

Figure 5 – Dataset is divided into two periods and these periods are further split into train/test sets.

Only after the intervals are defined and split into train and test set that is possible to do feature normalization. This step is done only after splitting the dataset to avoid data leakage where unseen future datapoints could influence the data standardization in a negative way. The standard scaler was applied to each of the features where the mean is subtracted from the value and split by its standard deviation.

3.5. Data Exploration

To understand the shape of the data under analysis, figure 7 presents graphically the behavior of the data along of the period studied. Is possible to spot that BTC price has just reached its all-time high in 2024. The MACD chart shows several crossovers above and below the signal line, corresponding to bullish and bearish trends, respectively. The strong upward momentum observed in 2017 and 2021 reflects significant price rallies, while the crossovers in 2018 and mid-2021 indicate market corrections.

The width of the Bollinger Bands widens during periods of high volatility, such as the price spikes in 2017 and 2021, and contracts during more stable periods. This highlights the periods of high market activity and subsequent price volatility. Additionally, the growth in unique addresses over time suggests increasing adoption and user interest in Bitcoin. The sharp rise in unique addresses during major bull runs indicates a surge in new participants entering the market.

The hash rate has shown a consistent upward trend, reflecting growing investment in mining infrastructure and increased network security. The steady rise in network difficulty aligns with the increasing hash rate, indicating enhanced security and more competition among miners. This trend underscores the growing computational effort required to mine new blocks.

S&P 500, Gold, and Oil Prices show steady growth with periodic corrections. The S&P 500 and gold have generally trended upward, while oil prices exhibit more volatility, particularly with the 2020 crash due to the pandemic.

The spike in 2020 on Initial Claims for Unemployment and the sharp drop in 2020 on Federal funds rate are related to pandemic period and provide insights into global economic conditions.

The analysis of these charts suggests a detailed examination of Bitcoin's market dynamics within the broader financial and economic context, providing insights into the interplay between cryptocurrency markets and traditional financial systems.

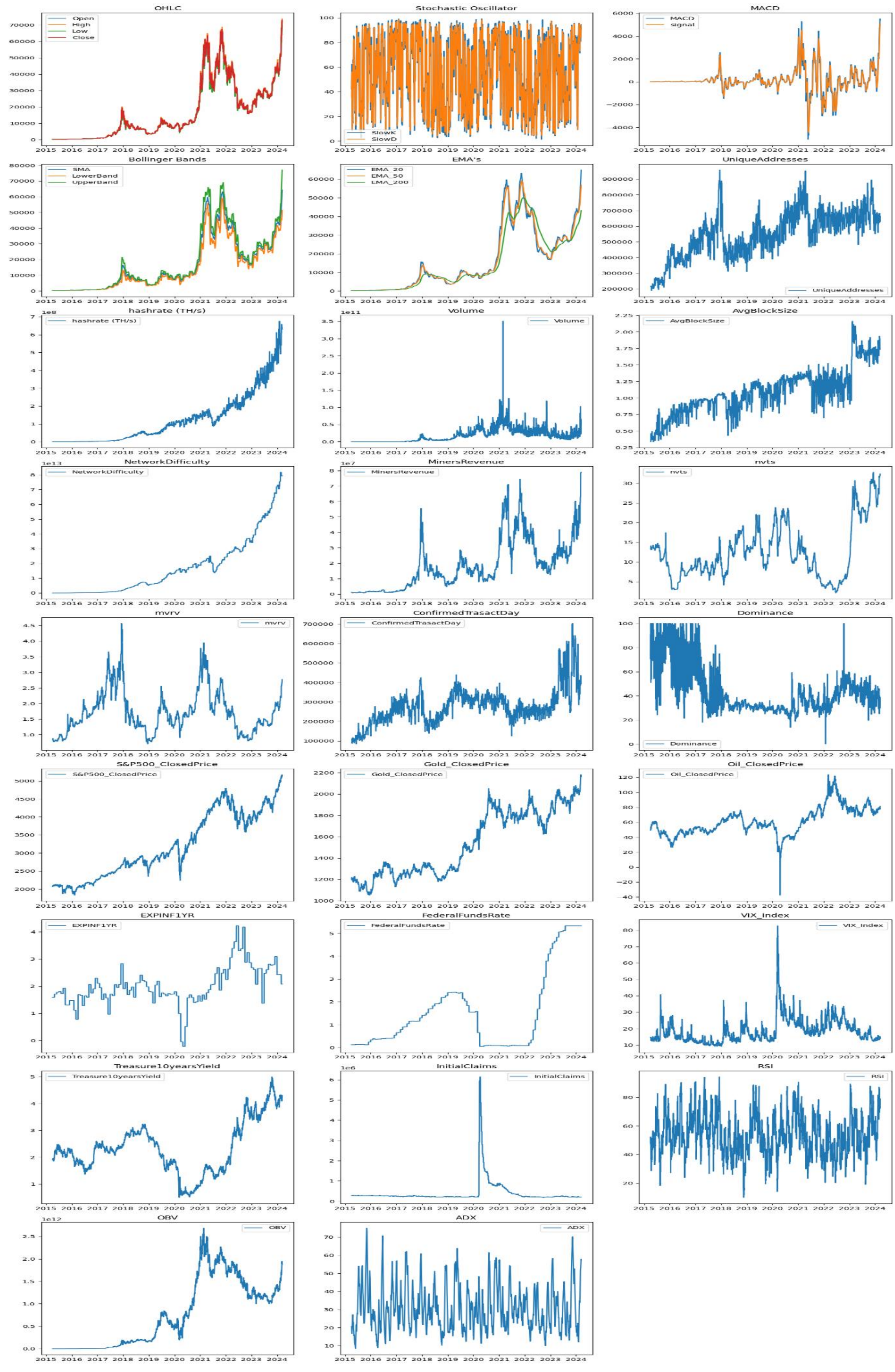
OHLC
Open
High
Low
Close
Stochastic Oscillator
SlowK
SlowD
MACD
MACD
signal
Bollinger Bands
SMA
LowerBand
UpperBand
EMA's
EMA_20
EMA_50
EMA_200
UniqueAddresses
UniqueAddresses
hashrate (TH/s)
hashrate (TH/s)
Volume
Volume
AvgBlockSize
AvgBlockSize
NetworkDifficulty
NetworkDifficulty
MinersRevenue
MinersRevenue
nvts
nvts
mvrv
mvrv
ConfirmedTrasactDay
ConfirmedTrasactDay
Dominance
Dominance
S&P500_ClosedPrice
S&P500_ClosedPrice
Gold_ClosedPrice
Gold_ClosedPrice
Oil_ClosedPrice
Oil_ClosedPrice
EXPINF1YR
EXPINF1YR
FederalFundsRate
FederalFundsRate
VIX_Index
VIX_Index
Treasure10yearsYield
Treasure10yearsYield
InitialClaims
InitialClaims
RSI
RSI
OBV
OBV
ADX
ADX

Figure 6 - Variables Subplot.

3.6. MODELING

Using python libraries, the algorithms are trained on different training sets. For LSTM the python library used is Keras, while for PatchTST is NeuralForecast.

The LSTM algorithm can be defined as:

$$\boldsymbol{h_t} = \boldsymbol{o_t} \odot \tanh(\boldsymbol{c_t})$$

Where:

- $\boldsymbol{h_t}$ is the hidden state vector time at time t.
- $\boldsymbol{o_t}$ is the output gate activation vector at time t and can be calculated as follows:
$$\boldsymbol{o_t} = \sigma(W_{xo^{xt}} + W_{ho^{h_{t-1}}} + W_{co^{c_{t-1}}} + b_o)$$
- $\odot$ represents element-wise multiplication.
- tanh is the hyperbolic tangent function.
- $\boldsymbol{c_t}$ is the cell state vector at time t. Can be calculated as:
$$\boldsymbol{c_t} = \boldsymbol{f_t} \odot \boldsymbol{c_{t-1}} + \boldsymbol{i_t} \odot \tanh\left(\boldsymbol{W_{xc^{h_t}}} + \boldsymbol{W_{hc^{h_{t-1}}}} + b_c\right)$$
- $\boldsymbol{f_t}$ is the forget gate activation vector at time t and can be calculated as follows:
$$\boldsymbol{f_t} = \sigma(W_{xi^{xt}} + W_{hf^{h_{t-1}}} + W_{cf^{c_{t-1}}} + b_f)$$
- $\boldsymbol{i_t}$ is the input gate activation vector at time t and can be calculated as follows:
$$\boldsymbol{i_t} = \sigma(W_{xi^{xt}} + W_{hi^{h_{t-1}}} + W_{ci^{c_{t-1}}} + b_i)$$
- $\boldsymbol{x_t}$ is the input vector at time t.
- $\boldsymbol{h_{t-1}}$ is the hidden state vector from the previous time step.
- W_{xo} is the weight matrix that connects the input vector x_t to the output gate o_t.
- W_{ho} is the weight matrix that connects the hidden state vector h_{t-1} from previous time step to the output gate o_t.
- W_{co} is the weight matrix that connects the cell state vector $c_t - 1$ from the previous time step to the output gate o_t.
- b_o, b_c, b_f, b_i are bias vector parameters.
- σ is the sigmoid function.

The PatchTST algorithm is a multi-step algorithm, hence the steps can be defined as:

Patching:

Each input univariate time series, $x^{(i)}$, is divided into patches. Let *P* be the patch length and *S* be the stride. The number of patches generated, *N*, can be calculated as follows:

$$N = \left\lceil \frac{(L-P)}{S} \right\rceil + 2$$

Linear Projection and Position Encoding:

Each patch, $x_p^{(i)}$, is mapped to the Transformer latent space of dimension D via a trainable linear projection matrix W_p of size D x P. A learnable additive position encoding W_{pos} of size D x N is applied to monitor the temporal order of patches. The input fed into the Transformer encoder is denoted as $x_d^{(i)}$ and is calculated as:

$$x_d^i = W_{p^{x_p^{(i)}}} + W_{pos}$$

Multi-Head Attention:

For each head h, multi-head attention transforms $x_d^{(i)}$ into query matrices $Q_h^{(i)}$, key matrices $K_h^{(i)}$, and value matrices $V_h^{(i)}$ as follows:

$$Q_h^{(i)} = (x_d^{(i)})^T W_h^k, \qquad K_h^{(i)} = (x_d^{(i)})^T W_h^k, \qquad V_h^{(i)} = (x_d^{(i)})^T W_h^k$$

The attention output $O_h^{(i)}$ is obtained using the scaled dot-product attention mechanism:

$$O_h^{(i)} = Attention(Q_h^{(i)}, K_h^{(i)}, V_h^{(i)} = Softmax\left(\frac{Q_h^{(i)}(K_h^{(i)})^T}{\sqrt{d_k}}\right) V_h^{(i)}$$

Multi-Head Attention Block:

After obtaining the attention output $O_h^{(i)}$, the multi-head attention block also includes BatchNorm layers and a feed-forward network with residual connections. The output is denoted as $z^{(i)}$.

Prediction:

After the multi-head attention, a flatten layer with a linear head is used to obtain the prediction result $\hat{x}$:

$$\hat{x}^{(i)} = W_{linear^{z^{(i)}}}$$

3.7. Model Training and Tuning

Since PatchTST authors stressed the accuracy of the model on long horizons, the models are trained, evaluated, and compared throughout different looking back windows and horizons, therefore, is possible to evaluate the performance on long- and short-term forecast. A similar approach was taken in the study of (Dutta et al., 2020) where the lookbacks windows studied were 15, 30, 45 and 60 to forecast one day ahead of Bitcoin prices. Additionally, the study of (Pichaiyuth et al., 2023), studied the forecast performance on 7, 15, 30, 60 and 90 days ahead horizon. It was tested a different range of lookback window in the given horizon set, only the best performance was chosen. The lookback window and horizon used in this study are shown in table 3:

Table 3 – Lookback and horizon scenarios

Lookback Window	Horizon
14	7
30	14
60	30
105	45
165	75
210	90

To improve the model's forecast results a set of different parameters were tested. The tests were done by try and error based on literature review previously done. For this study it was considered the number of epochs, number of units, and batch size. The range of number of number of epochs tested were 50, 100, 200, and 250, for number of units the range was 32, 64, 128 and 256 and for batch size was 16, 32, 64, and 128. This range of parameters were employed in the study of da Silva & Meneses, 2023; Dutta et al., 2020 and Hua, 2020. The models were evaluated, and the best model was chosen.

The configuration of the model chosen have a selected loss function as Mean Average Error (MAE). The number of epochs was set to 200 and batch size is set to 32 on both models. The LSTM model was set to have two LSTM layers, the first with return sequences activated and having 64 units, the second with return sequences disable and with 32 units. Lastly is set a Dense layer followed by a Reshape layer. For PatchTST the tunning was done considering the number of head equal to 8 and hidden size to be 8, patch length was set to 8, the same as describe in PatchTST paper.

3.8. Evaluation Metrics

To evaluate the accuracy of both models the measures used were MAE (Mean Average Error), RMSE (Root Mean Squared Error), and MAPE (Mean Absolute Percentage Error). As (Hyndman & Koehler, 2006) explain, MAE measures the average absolute error between the expected value and the predicted value, while RSME measures the standard deviation, lastly, MAPE measures the absolute magnitude of error. The measures can be expressed as:

$$MAE = \left(\frac{1}{n}\right)\sum_{t=1}^{n} |d_t - z_t|$$

$$RSME = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(d_t - z_t)^2}$$

$$MAPE = \frac{100}{n}\sum_{t=1}^{n}\left|\frac{(d_t - z_t)}{d_t}\right|$$

The e_t measure is the forecast error calculated as $(d_t - z_t)$, d_t is the actual Bitcoin price at time t, z_t is the predicted value at time t, n is the total number of data points.

4. RESULTS AND DISCUSSION

After pre-processing, splitting, training, and generating predictions is possible to evaluate the models. In this section, a comparative analysis of the algorithms is done on different lookback and horizon periods. As mentioned before, the evaluation metrics employed include MAE, MAPE and RMSE and they are shown for period 1 and period 2. Table 4 shows the performance of the models.

Table 4 – Performance Metrics for LSTM and PatchTST

Horizon	Lookback	Metric	Period 1 LSTM	Period 1 PatchTST	Period 2 LSTM	Period 2 PatchTST
1	3	Mean Absolute Error (MAE)	149.72	**10.36**	**59.03**	64.81
		Mean Absolute Percentage Error (MAPE)	2.25%	**0.16%**	**0.20%**	0.22%
		Root Mean Squared Error (RMSE)	149.72	**10.36**	**59.03**	64.81
7	14	Mean Absolute Error (MAE)	232.33	**178.57**	953.13	**370.59**
		Mean Absolute Percentage Error (MAPE)	3.69%	**2.84%**	3.27%	**1.30%**
		Root Mean Squared Error (RMSE)	287.36	**229.16**	1 193.40	**534.29**
14	30	Mean Absolute Error (MAE)	433.29	**174.14**	1 349.12	**1 117.42**
		Mean Absolute Percentage Error (MAPE)	6.75%	**2.74%**	4.80%	**4.01%**
		Root Mean Squared Error (RMSE)	469.64	**225.02**	1 543.07	**1 499.09**
30	60	Mean Absolute Error (MAE)	356.52	**182.71**	**1 345.06**	1 394.41
		Mean Absolute Percentage Error (MAPE)	5.56%	**2.84%**	**4.93%**	4.99%
		Root Mean Squared Error (RMSE)	413.04	**210.32**	**1 521.79**	1 642.73
45	105	Mean Absolute Error (MAE)	543.90	**541.85**	**658.93**	1 041.20
		Mean Absolute Percentage Error (MAPE)	10.29%	**9.32%**	**2.42%**	3.75%
		Root Mean Squared Error (RMSE)	931.12	**651.05**	**816.99**	1 297.26
75	150	Mean Absolute Error (MAE)	**1 247.26**	1 225.45	3 657.34	**2 733.46**
		Mean Absolute Percentage Error (MAPE)	**30.45%**	30.75%	13.49%	**9.25%**
		Root Mean Squared Error (RMSE)	**1 544.14**	1 599.65	4 264.26	**4 085.33**
90	210	Mean Absolute Error (MAE)	2 037.61	**1 449.03**	**2 251.60**	3 429.18
		Mean Absolute Percentage Error (MAPE)	49.45%	**37.70%**	**7.80%**	11.60%
		Root Mean Squared Error (RMSE)	2 310.41	**1 909.96**	**2 905.23**	4 533.95

Note. Bold values indicate the best performance. MAE and RSME are given in USD.

Horizon	Lookback	Metric	Period 1 LSTM	Period 1 PatchTST	Period 2 LSTM

1	3	Mean Absolute Error (MAE)	149.72	10.36	59.03	6
		Mean Absolute Percentage Error (MAPE)	2.25%	0.16%	0.20%	
		Root Mean Squared Error (RMSE)	149.72	10.36	59.03	6
7	14	Mean Absolute Error (MAE)	232.33	370.59	953.13	3
		Mean Absolute Percentage Error (MAPE)	3.69%	1.30%	3.27%	
		Root Mean Squared Error (RMSE)	287.36	534.29	1 193.40	5
14	30	Mean Absolute Error (MAE)	433.29	174.14	1 349.12	1
		Mean Absolute Percentage Error (MAPE)	6.75%	2.74%	4.80%	
		Root Mean Squared Error (RMSE)	469.64	225.02	1 543.07	4
30	60	Mean Absolute Error (MAE)	356.52	182.71	1 345.06	3
		Mean Absolute Percentage Error (MAPE)	5.56%	2.84%	4.93%	
		Root Mean Squared Error (RMSE)	413.04	210.32	1 521.79	6
45	105	Mean Absolute Error (MAE)	543.90	541.85	658.93	0
		Mean Absolute Percentage Error (MAPE)	10.29%	9.32%	2.42%	
		Root Mean Squared Error (RMSE)	931.12	651.05	816.99	2
75	150	Mean Absolute Error (MAE)	1 247.26	1 225.45	3 657.34	7
		Mean Absolute Percentage Error (MAPE)	30.45%	30.75%	13.49%	
		Root Mean Squared Error (RMSE)	1 544.14	1 599.65	4 264.26	0
90	210	Mean Absolute Error (MAE)	2 037.61	1 449.03	2 251.60	4
		Mean Absolute Percentage Error (MAPE)	49.45%	37.70%	7.80%	
		Root Mean Squared Error (RMSE)	2 310.41	1 909.96	2 905.23	5

Observing the evaluation metrics is possible to identify that PatchTST outperforms LSTM not only on long horizon forecast, but in short-term prediction as well. PatchTST demonstrates substantially lower errors, suggesting a better capability in capturing short-term dependencies and variations in the data. As the lookback and horizon increases the error rate increases, however not at the same pace as the compared algorithm, indicating that the Transformer based algorithm has a better capability in handling long term dependencies than LSTM.

For one day ahead prediction, both models produced low errors. For period 1 PatchTST could better predict the next step than LSTM, however for period 2 the performance was very similar. For horizon 7 and lookback window 14, the Transformer algorithm outperforms LSTM on all metrics, indicating that on average, PatchTST produces more accurate predictions when the period considered is the previous two weeks, however, is important to mentioned that for period 1 the difference on MAPE and RSME was very low, but for period 2 LSTM had more than the double of PatchTST RMSE.

For mid-term predictions, with a horizon from 14 to 45 and lookback from 30 to 105, PatchTST had a better performance on period 1. On period 2 the accuracy of both models are very similar, indicating an equal capability of capturing the variance of the data. On long-term prediction is where the MAPE and RSME had significantly increased on both models, indicating that the models had problems on capturing the behavior of price movements. For period 1, the best performance was LSTM for horizon 75 and PatchTST for horizon 90, while for period 2 was the opposite.

When the results are plotted in the periods, is possible to identify the differences found in the evaluation metrics. Figures 7 and 8 are the plot results.

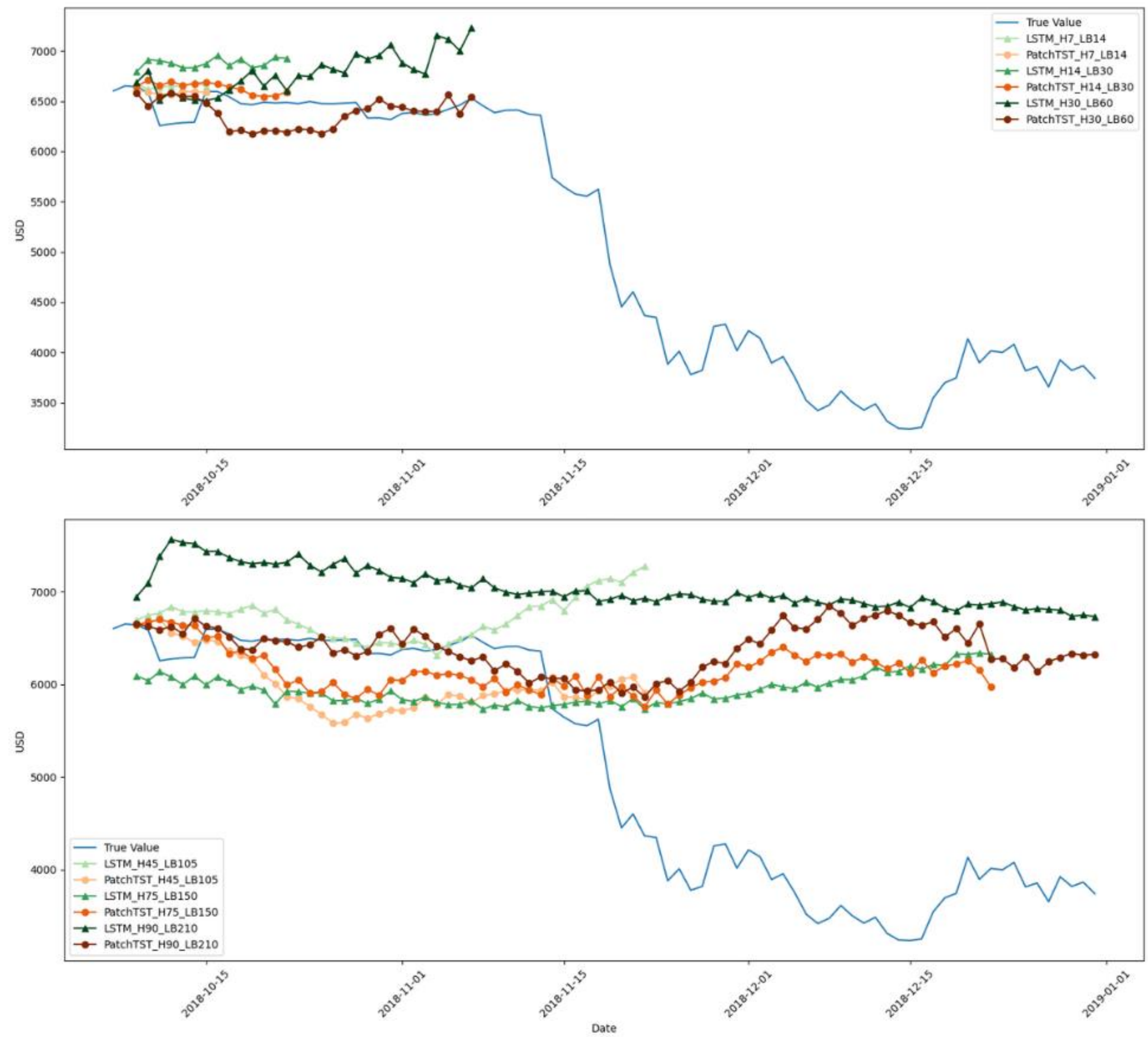

Figure 7 - Period 1 Plot Results

On the first window forecast, none of the models could predict the downtrend that starts on 13/11/2018, demonstrating that both algorithms have problems on capturing patterns on long horizon and lookback period as it can be seen on LSTM_H90_LB210 and PatchTST_H90_LB210. In the same horizon and lookback, however, PatchTST could maintain proximity to the true value until the beginning of the downtrend in mid-November.

When the results of period 1 are compared to simple naïve forecast, in other words, repeating the last observed value for the horizons studied, PatchTST could beat the simpler method in almost every horizon studied. The same cannot be said for LSTM, who performed poorly compared to a naïve forecast. Table 5 are the metrics results.

Table 5 – Evaluation Metrics Comparison to Naïve Forecast

Horizon	Metric	Period 1		
		LSTM	PatchTST	Naive Forecast
1	Mean Absolute Error (MAE)	149.72	10.36	**9.59**
	Mean Absolute Percentage Error (MAPE)	2.25%	0.16%	**0.14%**
	Root Mean Squared Error (RMSE)	149.72	10.36	**9.59**
7	Mean Absolute Error (MAE)	232.33	**178.57**	233.31
	Mean Absolute Percentage Error (MAPE)	3.69%	**2.84%**	3.70%
	Root Mean Squared Error (RMSE)	287.36	**229.16**	285.79
14	Mean Absolute Error (MAE)	433.29	**174.14**	189.81
	Mean Absolute Percentage Error (MAPE)	6.75%	**2.74%**	2.98%
	Root Mean Squared Error (RMSE)	469.64	**225.02**	229.11
30	Mean Absolute Error (MAE)	356.52	**182.71**	210.16
	Mean Absolute Percentage Error (MAPE)	5.56%	**2.84%**	3.29%
	Root Mean Squared Error (RMSE)	413.04	**210.32**	233.57
45	Mean Absolute Error (MAE)	543.90	541.85	**472.59**
	Mean Absolute Percentage Error (MAPE)	10.29%	9.32%	**8.81%**
	Root Mean Squared Error (RMSE)	931.12	651.05	**741.44**
75	Mean Absolute Error (MAE)	**1 247.26**	1 225.45	1 439.44
	Mean Absolute Percentage Error (MAPE)	**30.45%**	30.75%	36.55%
	Root Mean Squared Error (RMSE)	**1 544.14**	1 599.65	1 926.89
90	Mean Absolute Error (MAE)	2 037.61	**1 449.03**	1 662.00
	Mean Absolute Percentage Error (MAPE)	49.45%	**37.70%**	42.41%
	Root Mean Squared Error (RMSE)	2 310.41	**1 909.96**	2 092.69

Note. Bold values indicate the best performance. MAE and RSME are given in USD.

In short term predictions like 1 day horizon, both PatchTST and the Naive forecast significantly outperform the LSTM model, with RSME values of 10.36 and 9.59 respectively, compared to LSTM's 149.72. The MAPE and RMSE follow a similar pattern. As the forecasting horizon increases to 7 and 14 days, PatchTST continues to show lower errors across all metrics compared to LSTM and Naive forecasts. For instance, at the 14-day horizon, PatchTST's MAPE is 2.74%, significantly lower than LSTM's 6.75%, and still better than the Naive forecast's 2.98%. This trend indicates that PatchTST maintains its robustness over medium-term periods better than LSTM and Naive approaches.

For longer horizons, however, the performance gap between these models starts to narrow, and in some cases, naïve forecasts perform comparably or better than the more sophisticated models. At the 45-day horizon, the MAE for the Naive forecast is 472.59, slightly lower than LSTM's 543.90 and almost equivalent to PatchTST's 541.85.

Figure 8 plots the results of the predicted value and the true value for the period 2 as follows:

Figure 7 – Period 2 Plot Results

The reversion of a downtrend that occurred on 14/06/2023 could only be predicted by LSTM on LSTM_H90_LB210, however with a lag of around 6 days. Apart from that, neither of the lookbacks nor horizon could predict this reversion. On horizon 75, LSTM had a poor performance while PatchTST could maintain a close distance to the true value until the reversion trend in mid July 7. Period 2 was where LSTM performed better than PatchTST more often, however between the MAPE and the RSME when this happened was not big. This indicates that the models had a similar capability on understand the dependencies of the data.

The metrics of both models are again compared to a naïve forecast. The results are shown in table 6.

Table 6 – Evaluation Metrics Comparison to Naïve Forecast

Horizon	Metric	Period 2		
		LSTM	PatchTST	Naive Forecast
1	Mean Absolute Error (MAE)	**59.03**	64.81	133.53
	Mean Absolute Percentage Error (MAPE)	**0.20%**	0.22%	0.46%
	Root Mean Squared Error (RMSE)	**59.03**	64.81	133.53
7	Mean Absolute Error (MAE)	953.13	**370.59**	547.55
	Mean Absolute Percentage Error (MAPE)	3.27%	**1.30%**	1.91%
	Root Mean Squared Error (RMSE)	1 193.40	**534.29**	682.53
14	Mean Absolute Error (MAE)	1 349.12	1 117.42	**957.21**
	Mean Absolute Percentage Error (MAPE)	4.80%	4.01%	**3.43%**
	Root Mean Squared Error (RMSE)	1 543.07	1 499.09	**1 214.00**
30	Mean Absolute Error (MAE)	**1 345.06**	1 394.41	1 825.08
	Mean Absolute Percentage Error (MAPE)	**4.93%**	4.99%	6.73%
	Root Mean Squared Error (RMSE)	**1 521.79**	1 642.73	2 070.41
45	Mean Absolute Error (MAE)	**658.93**	1 041.20	2 082.60
	Mean Absolute Percentage Error (MAPE)	**2.42%**	3.75%	7.73%
	Root Mean Squared Error (RMSE)	**816.99**	1 297.26	2 294.69
75	Mean Absolute Error (MAE)	3 657.34	2 733.46	**1 895.98**
	Mean Absolute Percentage Error (MAPE)	13.49%	9.25%	**6.97%**
	Root Mean Squared Error (RMSE)	4 264.26	4 085.33	**2 173.29**
90	Mean Absolute Error (MAE)	2 251.60	3 429.18	**1 682.17**
	Mean Absolute Percentage Error (MAPE)	7.80%	11.60%	**6.15%**
	Root Mean Squared Error (RMSE)	2 905.23	4 533.95	**2 007.80**

Note. Bold values indicate the best performance. MAE and RSME are given in USD.

Comparing results on period 2 to a naïve forecast, LSTM exhibits the lowest errors with an MAE of 59.03 and a MAPE of 0.20%, significantly outperforming both PatchTST and the Naive forecast for 1-day horizon forecast. However, at the 14-day horizon, the performance dynamics shift. The LSTM and PatchTST models show closer error values, with LSTM having a higher MAE (1 349.12) compared to PatchTST (1 117.42) but a slightly better MAPE (4.80% versus 4.01%). Interestingly, the Naive forecast shows lower MAE (957.21) and MAPE (3.43%) than both models, indicating poor performance for this mid-term horizon.

At long horizons of 75 and 90 days, the models show divergent performance patterns, with Naive forecast being the most stable, indicating that a simple method could perform better than robust models.

Overall, PatchTST demonstrates a clear advantage over LSTM across various horizons and lookback periods. The results consistently indicate that PatchTST outperforms LSTM not only in short-term predictions but also in capturing long-term dependencies and variations in the data. This is evident in the lower error metrics (MAE, MAPE, and RMSE) across most tested scenarios, particularly in Period 1.

The superior performance indicates that the architecture of the Transformers based algorithm captures better, on average, the short-term and long-term dependencies than LSTM. As the forecast horizon increases, the error increases, however is less noticeable in PatchTST than in the LSTM, which confirms the study of (Nie et al., 2023) whereas the PatchTST had the best performance on predicting long horizon time series.

However, it's important to note that for certain long-term horizons, particularly in Period 2, the performance gap between PatchTST and LSTM descreases, and in some cases, LSTM outperformed PatchTST. This indicates that while PatchTST generally excels, its advantage is not uniformly consistent across all conditions and periods.

The implications of a naive forecast outperforming robust algorithms like LSTM and PatchTST are significant. It suggests that despite the sophistication of LSTM and PatchTST, there are scenarios where simpler forecasting methods are more reliable. The naive forecast, requiring minimal computational resources, highlights potential limitations in the robustness of LSTM and PatchTST for extended periods. This emphasizes the importance of evaluating model performance against simple benchmarks to ensure the increased computational effort is justified. Moreover, it suggests that for certain forecasting tasks, especially those involving longer horizons, simpler models might provide more reliable results.

Therefore, the results of the present study fulfill the proposed objectives: (i) assess the accuracy and reliability of the PatchTST algorithm in predicting Bitcoin price fluctuations using technical, macroeconomic, and blockchain indicators; (ii) Conduct a comparative analysis by benchmarking the PatchTST algorithm against the LSTM algorithm.

CONCLUSIONS AND FUTURE WORKS

The growing adoption of Bitcoin makes it crucial to understand its prices movement. Additionally, the recent advancements in Natural Processing Language with GPT models have created the possibility to apply the same architecture for time series forecasting. The new technologies must be applied and tested in a variety of problems, including Bitcoin forecasting.

In this sense, this study systematically compares the Transformer-based algorithm, PatchTST against the RNN-based algorithm, LSTM on the problem of Bitcoin forecasting. The models were trained and evaluated in different time frames, additionally they were trained in different lookback and horizons windows. The metrics are then averaged to create a final output for each horizon and lookback period.

The results show that the Transformers-based algorithm has outperformed the LSTM model in short- and long-term forecasting. The lower MAPE and RSME in the majority of the cases, indicate that PatchTST forecasts values closer to the true value than LSTM, and the variance of the error is of PatchTST is lower than LSTM.

Regarding the limitations of the present study and possible future work, the use of different sets of variables could play a role in forecasting Bitcoin as it could capture better the correlation between price fluctuations. The use of a different data frequency could also be used. In this study it was used a daily frequency of the data points, and perhaps a different use of timestamp could benefit the process of forecasting. Lastly, the use of different variants of LSTM should be considered, for example BiLSTM and ConvLSTM as the work of (da Silva & Meneses, 2023) suggests a better performance of this algorithms over LSTM on power consumption dataset.

BIBLIOGRAPHICAL REFERENCES

Aalborg, H. A., Molnár, P., & de Vries, J. E. (2019). What can explain the price, volatility and trading volume of Bitcoin? *Finance Research Letters*, *29*, 255–265. https://doi.org/10.1016/j.frl.2018.08.010

Aggarwal, A., Gupta, I., Garg, N., & Goel, A. (2019). Deep Learning Approach to Determine the Impact of Socio Economic Factors on Bitcoin Price Prediction. *2019 Twelfth International Conference on Contemporary Computing (IC3)*, 1–5. https://doi.org/10.1109/IC3.2019.8844928

Atsalakis, G. S., Atsalaki, I. G., Pasiouras, F., & Zopounidis, C. (2019). Bitcoin price forecasting with neuro-fuzzy techniques. *European Journal of Operational Research*, *276*(2), 770–780. https://doi.org/10.1016/j.ejor.2019.01.040

Basher, S., & Sadorsky, P. (2022). Forecasting Bitcoin price direction with random forests: How important are interest rates, inflation, and market volatility? *Machine Learning with Applications*, *9*(6), 100355. https://doi.org/10.1016/j.mlwa.2022.100355

Box, G., Jenkins, G., Reinsel, G., & Ljung, G. (2015). *Time Series Analysis: Forecasting and Control* (5th ed.). John Wiley & Sons.

Chen, J. (2023). Analysis of Bitcoin Price Prediction Using Machine Learning. *Journal of Risk and Financial Management*, *16*(1), 51. https://doi.org/10.3390/jrfm16010051

Chen, W., Xu, H., Jia, L., & Gao, Y. (2021). Machine learning model for Bitcoin exchange rate prediction using economic and technology determinants. *International Journal of Forecasting*, *37*(1), 28–43. https://doi.org/10.1016/j.ijforecast.2020.02.008

Chen, Z., Li, C., & Sun, W. (2020). Bitcoin price prediction using machine learning: An approach to sample dimension engineering. *Journal of Computational and Applied Mathematics*, *365*, 112395. https://doi.org/10.1016/j.cam.2019.112395

Cocco, L., Tonelli, R., & Marchesi, M. (2021). Predictions of bitcoin prices through machine learning based frameworks. *PeerJ Computer Science*, *7*, e413. https://doi.org/10.7717/peerj-cs.413

da Silva, D. G., & Meneses, A. A. de M. (2023). Comparing Long Short-Term Memory (LSTM) and bidirectional LSTM deep neural networks for power consumption prediction. *Energy Reports*, *10*, 3315–3334. https://doi.org/10.1016/j.egyr.2023.09.175

Dutta, A., Kumar, S., & Basu, M. (2020). A Gated Recurrent Unit Approach to Bitcoin Price Prediction. *Journal of Risk and Financial Management*. https://doi.org/10.3390/jrfm13020023

Erfanian, S., Zhou, Y., Razzaq, A., Abbas, A., Safeer, A. A., & Li, T. (2022). Predicting Bitcoin (BTC) Price in the Context of Economic Theories: A Machine Learning Approach. *Entropy*, *24*(10), Article 10. https://doi.org/10.3390/e24101487

Goodell, J. W., Ben Jabeur, S., Saâdaoui, F., & Nasir, M. A. (2023). Explainable artificial intelligence modeling to forecast bitcoin prices. *International Review of Financial Analysis*, *88*, 102702. https://doi.org/10.1016/j.irfa.2023.102702

Graves, A., & Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, *18*(5), 602–610. https://doi.org/10.1016/j.neunet.2005.06.042

Gujarati, D. N., & Porter, D. C. (2009). *Basic econometrics* (5. ed). McGraw-Hill Irwin.

Hewamalage, H., Bergmeir, C., & Bandara, K. (2021). Recurrent Neural Networks for Time Series Forecasting: Current status and future directions. *International Journal of Forecasting*, *37*(1), 388–427. https://doi.org/10.1016/j.ijforecast.2020.06.008

Hochreiter, S. (1998). The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, *6*(2). https://doi.org/10.1142/S0218488598000094

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-term Memory. *Neural Computation*, *9*, 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Hua, Y. (2020). Bitcoin price prediction using ARIMA and LSTM. *E3S Web of Conferences*, *218*, 01050. https://doi.org/10.1051/e3sconf/202021801050

Huang, J.-Z., Huang, W., & Ni, J. (2019). Predicting bitcoin returns using high-dimensional technical indicators. *The Journal of Finance and Data Science*, *5*(3), 140–155. https://doi.org/10.1016/j.jfds.2018.10.001

Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, *22*(4), 679–688. https://doi.org/10.1016/j.ijforecast.2006.03.001

Jabbar, M. A., Abraham, A., Dogan, O., Madureira, A. M., & Tiwari, S. (2021). *Deep Learning in Biomedical and Health Informatics: Current Applications and Possibilities*. CRC Press.

Khan, S., Naseer, M., Hayat, M., Zamir, S. W., Khan, F. S., & Shah, M. (2022). Transformers in Vision: A Survey. *ACM Computing Surveys*, *54*(10s), 200:1-200:41. https://doi.org/10.1145/3505244

Kontopoulou, V. I., Panagopoulos, A. D., Kakkos, I., & Matsopoulos, G. K. (2023). A Review of ARIMA vs. Machine Learning Approaches for Time Series Forecasting in Data Driven Networks. *Future Internet*, *15*(8), Article 8. https://doi.org/10.3390/fi15080255

Latif, N., Selvam, J. D., Kapse, M., Sharma, V., & Mahajan, V. (2023). Comparative Performance of LSTM and ARIMA for the Short-Term Prediction of Bitcoin Prices. *Australasian Accounting, Business and Finance Journal*, *17*(1), 256–276. https://doi.org/10.14453/aabfj.v17i1.15

Lim, B., Arık, S. Ö., Loeff, N., & Pfister, T. (2021). Temporal Fusion Transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, *37*(4), 1748–1764. https://doi.org/10.1016/j.ijforecast.2021.03.012

Lin, T., Wang, Y., Liu, X., & Qiu, X. (2022). A survey of transformers. *AI Open*, *3*, 111–132. https://doi.org/10.1016/j.aiopen.2022.10.001

Lopes, E., & Bianchi, R. (2022). Short-term prediction for Ethereum with Deep NeuralNetworks. *Anais Do I Brazilian Workshop on Artificial Intelligence in Finance*, *1*, 1–12. https://doi.org/doi:10.5753/bwaif.2022.222629

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018). Statistical and Machine Learning forecasting methods: Concerns and ways forward. *PLOS ONE*, *13*(3), e0194889. https://doi.org/10.1371/journal.pone.0194889

McNally, S., Roche, J., & Caton, S. (2018). Predicting the Price of Bitcoin Using Machine Learning. *2018 26th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP)*, 339–343. https://doi.org/10.1109/PDP2018.2018.00060

Montgomery, D. C., Jennings, C. L., & Kulahci, M. (2015). *Introduction to Time Series Analysis and Forecasting*. John Wiley & Sons.

Munim, Z., Shakil, M., & Alon, I. (2019). Next-Day Bitcoin Price Forecast. *Journal of Risk and Financial Management*, *12*, 103. https://doi.org/10.3390/jrfm12020103

Murray, K., Rossi, A., Carraro, D., & Visentin, A. (2023). On Forecasting Cryptocurrency Prices: A Comparison of Machine Learning, Deep Learning, and Ensembles. *Forecasting*, *5*(1), 196–209. https://doi.org/10.3390/forecast5010010

Nagula, P. K., & Alexakis, C. (2022). A new hybrid machine learning model for predicting the bitcoin (BTC-USD) price. *Journal of Behavioral and Experimental Finance*, *36*, 100741. https://doi.org/10.1016/j.jbef.2022.100741

Nakamoto, S. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System*.

Nie, Y., Nguyen, N. H., Sinthong, P., & Kalagnanam, J. (2023). *A Time Series is Worth 64 Words: Long-term Forecasting with Transformers* (arXiv:2211.14730). arXiv. http://arxiv.org/abs/2211.14730

Parvez, S. J., R, A., R, B., & C, S. D. (2022). Bitcoin price prediction using Random Forest Regression. *Journal of Positive School Psychology*, *6*(4), 4352–4358.

Pichaiyuth, P., Termnuphan, P., Triyason, T., Rojanapornpun, O., & Jaiyen, S. (2023). Price Trend Forecasting of Cryptocurrency Using Multiple Technical Indicators and SHAP. *2023 20th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, 150–154. https://doi.org/10.1109/JCSSE58229.2023.10201984

Ramos Pérez, E., Alonso-González, P. J., & Nuñez-Velazquez, J. (2021). Multi-Transformer: A New Neural Network-Based Architecture for Forecasting S&P Volatility. *Mathematics*, *9*, 1794. https://doi.org/10.3390/math9151794

Rao, T. S., & Gabr, M. M. (1984). Tests for Linearity and Gaussianity of Stationary Time Series. In T. S. Rao & M. M. Gabr (Eds.), *An Introduction to Bispectral Analysis and Bilinear Time Series Models* (pp. 116–144). Springer US. https://doi.org/10.1007/978-1-4684-6318-7_4

Schäfer, A., & Zimmermann, H. (2006). *Recurrent neural networks are universal approximators*. 632–640.

Seabe, P., Moutsinga, C., & Pindza, E. (2023). Forecasting Cryptocurrency Prices Using LSTM, GRU, and Bi-Directional LSTM: A Deep Learning Approach. *Fractal Fract*, *7*(2), 203. https://doi.org/10.3390/fractalfract7020203

Serafini, G., Yi, P., Zhang, Q., Brambilla, M., Wang, J., Hu, Y., & Li, B. (2020). Sentiment-Driven Price Prediction of the Bitcoin based on Statistical and Deep Learning Approaches. *2020 International Joint Conference on Neural Networks (IJCNN)*, 1–8. https://doi.org/10.1109/IJCNN48605.2020.9206704

Shah, D., Isah, H., & Zulkernine, F. (2019). Stock Market Analysis: A Review and Taxonomy of Prediction Techniques. *International Journal of Financial Studies*, *7*, 1–22. https://doi.org/10.3390/ijfs7020026

Shin, M., Mohaisen, D., & Kim, J. (2021). Bitcoin Price Forecasting via Ensemble-based LSTM Deep Learning Networks. *2021 International Conference on Information Networking (ICOIN)*, 603–608. https://doi.org/10.1109/ICOIN50884.2021.9333853

Singh, E., Kuzhagaliyeva, N., & Sarathy, S. M. (2022). Using deep learning to diagnose preignition in turbocharged spark-ignited engines. In J. Badra, P. Pal, Y. Pei, & S. Som (Eds.), *Artificial Intelligence and Data Driven Optimization of Internal Combustion Engines* (pp. 213–237). Elsevier. https://doi.org/10.1016/B978-0-323-88457-0.00005-9

Staudemeyer, R. C., & Morris, E. R. (2019). *Understanding LSTM -- a tutorial into Long Short-Term Memory Recurrent Neural Networks* (arXiv:1909.09586). arXiv. http://arxiv.org/abs/1909.09586

Szetela, B., Mentel, G., & Gędek, S. (2016). Dependency Analysis between Bitcoin and Selected Global Currencies. *Dynamic Econometric Models*, *16*(1), 133. https://doi.org/10.12775/DEM.2016.009

Tandon, S., Tripathi, S., Saraswat, P., & Dabas, C. (2019). Bitcoin Price Forecasting using LSTM and 10-Fold Cross validation. *2019 International Conference on Signal Processing and Communication (ICSC)*, 323–328. https://doi.org/10.1109/ICSC45622.2019.8938251

Tiwari, R. G., Agarwal, A. K., Kaushal, R. K., & Kumar, N. (2021). Prophetic Analysis of Bitcoin price using Machine Learning Approaches. *2021 6th International Conference on Signal Processing, Computing and Control (ISPCC)*, 428–432. https://doi.org/10.1109/ISPCC53510.2021.9609419

Tripathi, B., & Sharma, R. K. (2022). Modeling Bitcoin Prices using Signal Processing Methods, Bayesian Optimization, and Deep Neural Networks. *Computational Economics*. https://doi.org/10.1007/s10614-022-10325-8

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is All you Need. *Advances in Neural Information Processing Systems*, *30*. https://papers.nips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

Wen, Q., Zhou, T., Zhang, C., Chen, W., Ma, Z., Yan, J., & Sun, L. (2023). Transformers in Time Series: A Survey. *32nd International Joint Conference on Artificial Intelligence*, 1–9. https://doi.org/10.48550/arXiv.2202.07125

Wirawan, I. M., Widiyaningtyas, T., & Hasan, M. M. (2019). Short Term Prediction on Bitcoin Price Using ARIMA Method. *2019 International Seminar on Application for Technology of Information and Communication (iSemantic)*, 260–265. https://doi.org/10.1109/ISEMANTIC.2019.8884257

Yen, K.-C., & Cheng, H.-P. (2021). Economic policy uncertainty and cryptocurrency volatility. *Finance Research Letters*, *38*, 101428. https://doi.org/10.1016/j.frl.2020.101428

Zeng, A., Chen, M., Zhang, L., & Xu, Q. (2023). Are Transformers Effective for Time Series Forecasting? *Proceedings of the AAAI Conference on Artificial Intelligence*, *37*(9), Article 9. https://doi.org/10.1609/aaai.v37i9.26317

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa