

Predictive Accuracy of Bitcoin Price Movements: A Comparative Analysis of Machine Learning Models and ARIMA

Golla Chakrapani¹, Somireddy Spandana², Vittapu Manisarma³

^{1,2}Assistant Professor, Department of CSE, Malla Reddy Engineering College and Management Sciences, Hyderabad, Telangana.

³Professor, Department of CSE, Malla Reddy Engineering College and Management Sciences, Hyderabad, Telangana.

Abstract

This paper focuses on assessing the accuracy of predicting the direction of Bitcoin's price in USD. Historical price data is extracted from the Bitcoin Price Index, and the task is approached with varying levels of success by leveraging Bayesian-optimized Support Vector Regression (SVR) methods and Long Short-Term Memory (LSTM) networks. Among the methods employed, LSTM achieves the highest classification accuracy at 52% and a Root Mean Square Error (RMSE) of 8%. In addition, the popular ARIMA model for time series forecasting is incorporated for comparison with machine learning models. As anticipated, the non-linear machine learning approaches outperform ARIMA, which exhibits poor performance. Lastly, the study includes a benchmarking analysis of both machine learning models implemented on both a GPU and a CPU, revealing a 67.7% improvement in training time for the GPU-based implementation.

Keywords: Bitcoin Price Prediction, Support Vector Regression, Long Short Term Memory, ARIMA, Machine Learning, GPU, CPU Benchmark.

I. INTRODUCTION

Bitcoin [1] is the world's most valuable cryptocurrency and is traded on over 40 exchanges worldwide accepting over 30 different currencies. It has a current market capitalization of 9 billion USD according to <https://www.blockchain.info/> and sees over 250,000 transactions taking place per day. As a currency, Bitcoin offers a novel opportunity for price prediction due to its relatively young age and resulting volatility, which is far greater than that of fiat currencies [2]. It is also unique in relation to traditional fiat currencies in terms of its open nature; no complete data exists regarding cash transactions or money in circulation for fiat currencies. Prediction of mature financial markets such as the stock market has been researched at length [3], [4]. Bitcoin presents an interesting parallel to this as it is a time series prediction problem in a market still in its transient stage. Traditional time series prediction methods such as Holt-Winters exponential smoothing models rely on linear assumptions and require data that can be broken down into trend, seasonal and noise to be effective [5]. This type of methodology is more suitable for a task such as forecasting sales where seasonal effects are present. Due to the lack of seasonality in the Bitcoin market and its high volatility, these methods are not very effective for this task. Given the complexity of the task, machine learning makes for an interesting technological solution

based on its performance in similar areas. The recurrent neural network (SVR) and the long short term memory (LSTM) are favoured over the traditional multilayer perceptron (MLP) due to the temporal nature of Bitcoin data. The aim of this paper is to investigate with what accuracy the price of Bitcoin can be predicted using machine learning and compare parallelisation methods executed on multi-core and GPU environments. This paper contributes in the following manner: of approximately 653 papers published on Bitcoin [6], only 7 (at the time of writing) are related to machine learning for prediction. To facilitate a comparison to more traditional approaches in financial forecasting, an ARIMA time series model is also developed for performance comparison purposes with the neural network models. The independent variable for this study is the closing price of Bitcoin in USD taken from the Coindesk Bitcoin Price Index. Rather than focusing on one specific exchange, we take the average price from five major Bitcoin exchanges: Bitstamp, Bitfinex, Coinbase, OkCoin and itBit. If we were to implement trades based on the signals it would be beneficial to focus on just one exchange. To assess the performance of models, we use the root mean squared error (RMSE) of the closing price and further encode the predicted price into categorical variable reflecting: price up, down or no change. This latter step allows for additional performance metrics that would be useful to a trader in the formation of a trading strategy: classification accuracy, specificity, sensitivity and precision. The dependent variables for this paper come from the Coindesk website, and Blockchain.info. In addition to the closing price, the opening price, daily high and daily low are also included as well as Blockchain data, i.e. the mining difficulty and hash rate. The features which have been engineered (considered as technical analysis indicators [7]) include two simple moving averages (SMA) and a de-noised closing price.

II. RELATED WORK

Research on predicting the price of Bitcoin using machine learning algorithms specifically is lacking. [8] implemented a latent source model as developed by [9] to predict the price of Bitcoin noting 89% return in 50 days with a Sharpe ratio of 4.1. There has also been work using text data from social media platforms and other sources to predict Bitcoin prices. [10] investigated sentiment analysis using support vector machines coupled with the frequency of Wikipedia views, and the network hash rate. [11] investigated the relationship between Bitcoin price, tweets and views for Bitcoin on Google Trends. [12] implemented a similar methodology except instead of predicting Bitcoin price they predicted trading volume using Google Trends views. However, one limitation of such studies is the often small sample size, and propensity for misinformation to spread through various (social) media channels such as Twitter or on message boards such as Reddit, which artificially inflate/deflate prices [13]. In the Bitcoin exchanges liquidity is considerably limited. As a result, the market suffers from a greater risk of manipulation. For this reason, sentiment from social media is not considered further. [14] analysed the Bitcoin Blockchain to predict the price of Bitcoin using support vector machines (SVM) and artificial neural networks (ANN) reporting price direction accuracy of 55% with a regular ANN. They concluded that there was limited predictability in Blockchain data alone. [15] also used Blockchain data, implementing SVM, Random Forests and Binomial GLM

(generalised linear model) noting prediction accuracy of over 97% however without cross-validating their models limiting the generalisability of their results. Wavelets have also been utilised to predict Bitcoin prices, with [16], [17] noting positive correlations between search engine views, network hash rate and mining difficulty with Bitcoin price. Building on these findings, data from the Blockchain, namely hash rate and difficulty are included in the analysis along with data from the major exchanges provided by CoinDesk. Predicting the price of Bitcoin can be considered analogous to other financial time series prediction tasks such as forex and stock prediction. Several bodies of research have implemented the MultiLayer Perceptron (MLP) for stock price prediction [4] [18]. However, the MLP only analyses one observation at a time [19]. In contrast, the output from each layer in a recurrent neural network (SVR) is stored in a context layer to be looped back in with the output from the next layer. In this sense, the network gains a memory of sorts as opposed to the MLP. The length of the network is known as the temporal window length. [20] notes that the temporal relationship of the series is explicitly modelled by the internal states contributing significantly to model effectiveness

III. METHODOLOGY

This paper follows the CRISP data mining methodology.¹ The motivation for CRISP-DM over the more traditional KDD [26] revolves around the business setting of the prediction task. The dataset Bitcoin dataset used, ranges from the 19th of August 2013 until the 19th of July 2016. A time series plot of this can be seen in Figure 1. Data from previous to August 2013 has been excluded as it no longer accurately represents the network. In addition to the Open, High, Low, Close (OHLC) data from CoinDesk, the difficulty and hash rate are taken from the Blockchain. The data was also standardised to give it a mean of 0 and standard deviation of 1. Standardisation was chosen over normalisation as it better suits the activation functions used by the machine learning models.

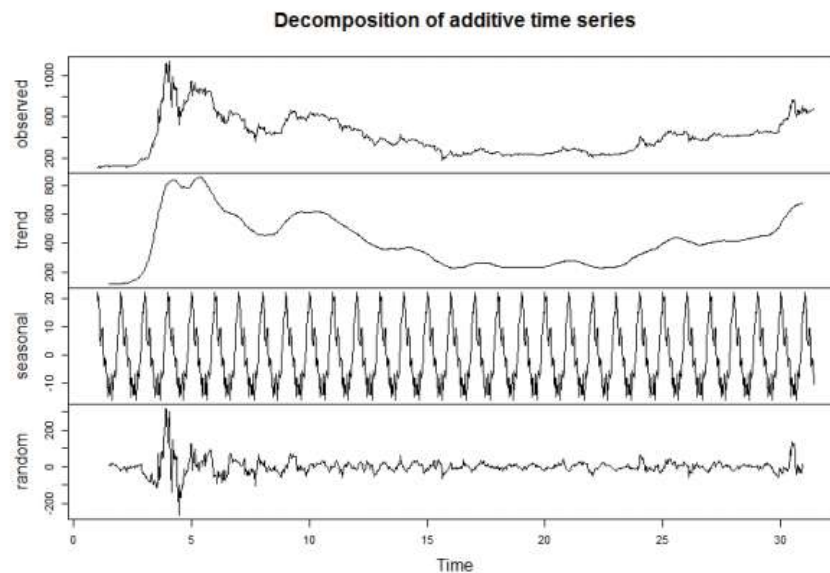


Figure 1: Decomposition of the Bitcoin Time Series Data

A. Feature Engineering and Feature Evaluation

Feature engineering is the art of extracting useful patterns from data to make it easier for machine learning models to perform their predictions. It can be considered one of the most important parts of the data mining process in order to achieve good results in prediction tasks. Several papers in recent years have included indicators including the Simple Moving Average (SMA) for machine learning classification tasks. An example of an appropriate technical indicator is a SMA recording the average price over the previous x days, and is correspondingly included.

To evaluate which features to include, Boruta (a wrapper built around the random forest classification algorithm) was used. This is an ensemble method in which classification is performed by voting of multiple classifiers. The algorithm works on a similar principle as the random forest classifier. It adds randomness to the model and collects results from the ensemble of randomised samples to evaluate attributes and provides a clear view on which attributes are important. All features were deemed important to the model based on the random forest, with 5 day and 10 days (via SMA) the highest importance among the tested averages. The de-noised closing price was one of the most important variables also.

B. Machine Learning Models

Appropriate design of machine learning models in terms of network parameters is imperative to their success. The three main options available when choosing how to select parameters for machine learning models are random search, grid search and heuristic search methods such as genetic algorithms. Manual grid search and Bayesian optimisation were utilised in this study. Grid search, implemented for the Elman SVR, is the process of selecting two hyperparameters with a minimum and maximum for each. One then searches that feature space looking for the best performing parameters. This approach was taken for parameters which were unsuitable for Bayesian optimisation. This model was built using Keras in the Python programming language. Similar to the SVR, Bayesian optimisation was chosen for selecting LSTM parameters where possible. This is a heuristic search method which works by assuming the function was sampled from a Gaussian process and maintains a posterior distribution for this function as the results of different hyperparameter selections are observed. One can then optimise the expected improvement over the best result to pick hyper parameters for the next experiment. The performance of both the SVR and LSTM network are evaluated on validation data with measures to prevent over fitting. Dropout is implemented in both layers, and we automatically stop model training if its validation loss hasn't improved in 5 epochs.

IV. IMPLEMENTATION

A. SVR

The first parameter to consider was the temporal length window. As suggested by supporting literature [12] these type of networks may struggle to learn long term dependencies using gradient based optimisation. An autocorrelation function (ACF) was run for the closing price

time series to assess the relationship between the current closing price and previous or future closing prices. While this is not a guarantee of predictive power for this length, it was a better choice than random choice. Closing price is correlated with a lag of up to 20 days in many cases, with isolated cases at 34, 45 and 47 days. This led the grid search for the temporal window to test from 2 to 20, 34, 45 and 47 days. To ensure a robust search, larger time periods of up to 100 days were also tested in increments of five. The most effective window temporal length was 24. In addition to the temporal window, some hyper parameters also need tuning: Learning rate is the parameter that guides stochastic gradient descent (SGD), i.e. how the network learns. Similarly, momentum updates the learning rate to avoid the model falling into local minima (in terms of error) and attempts to move towards the global minimum of the error function [35]. We used the RMSprop optimiser to improve on SGD, as it keeps a running average of recent gradients and as a result is more robust against information loss [36]. According to Heaton [37], one hidden layer is enough to approximate the vast majority of non-linear functions. Two hidden layers were also explored and were chosen as they achieved lower validation error. Heaton also recommends for the number of hidden nodes to select between the number of input and output nodes. In this case, less than 20 nodes per layer resulted in poor performance. 50 and 100 nodes were tested with good performance. However, too many nodes can increase the chances of overfitting, and significantly increase the time needed to train the network. As 20 nodes performed sufficiently well this was chosen for the final model. An activation function, a nonlinear stepwise equation that passes signals between layers, is also needed. The options explored were Tanh, ReLu, and Sigmoid. Tanh performed the best but the differences were not significant. The final parameters for selection are batch size and number of training epochs. Batch size was found to have little effect on accuracy but considerable effect on training time when using smaller batches in this case. The number of epochs tested ranged from 10 to 10000, however, too many training epochs can result in overfitting. To reduce the risk of overfitting, dropout was implemented as discussed above. Optimal dropout between 0.1 and 1 was searched for both layers with .5 dropout the optimal solution for both layers. A Keras callback method was also used to stop the training of the model if its performance on validation data did not improve after 5 epochs to prevent overfitting. Generally, the SVR converged between 20 and 40 epochs with early stopping.

B. LSTM

In terms of temporal length, the LSTM is considerably better at learning long term dependencies. As a result, picking a long window was less detrimental for the LSTM. This process followed a similar process to the SVR in which autocorrelation lag was used as a guideline. The LSTM performed poorly on smaller window sizes. Its most effective length found was 100 days, and two hidden LSTM layers were chosen. For a time series task two layers is enough to find nonlinear relationships among the data. 20 hidden nodes were also chosen for both layers as per the SVR model. The Hyperas library2 was used to implement the Bayesian optimisation of the network parameters. The optimiser searched for the optimal model in terms of how much dropout per layer and which optimizer to use. RMSprop again performed the best for this task. In the LSTM model, activation functions weren't changed as the LSTM has a particular sequence

of tanh and sigmoid activation functions for the different gates within the cell. LSTM models converged between 50 and 100 epochs with early stopping. Similar to the SVR, batch size was found to have a greater effect on execution time than accuracy. This may be due to the relatively small size of the dataset.

C. Model Comparison

A confusion matrix representing the ratio of true/false and positive/negative classifications is used to derive the ratings metrics. Accuracy can be defined as the total number of correctly classified predictions (price up, down, and no change). To combat inherent class imbalance (bitcoin price predominately increases) the metrics sensitivity, specificity and precision are also analysed. Sensitivity represents how good a model is at detecting positives. Specificity represent how good the model is at avoiding false alarms. Finally, precision represents how many positively classified predictions were relevant. Root Mean Square Error (RMSE) is used to evaluate and compare the regression accuracy. To instrument the evaluation of models, a 80/20 holdout validation strategy is used. In order to facilitate a comparison of the machine learning methods to more traditional methods we built (and optimised) an ARIMA model, as they have been extensively used in price prediction problems (e.g. [38], [39]). The ARIMA forecast was created by splitting the data into 5 periods and then predicting 30 days into the future. The data was differenced before being fit with several ARIMA models. The best fit was found by auto.arima from the R forecast package.

V. EVALUATION

LSTM achieved the highest accuracy while the SVR achieved the lowest RMSE. The ARIMA prediction performed poorly in terms of accuracy and RMSE. Upon analysis of the ARIMA forecast, it predicted the price would gradually rise each day. There were no false positives from the model. One reason for this may be due to the class imbalance in predictive portion of the ARIMA forecast (the price tends to always increase). This contributed to the specificity and precision being so high (specificity, precision= 100%). This does not necessarily suggest good overall performance, but rather that it does a decent job at identifying price direction change(s).

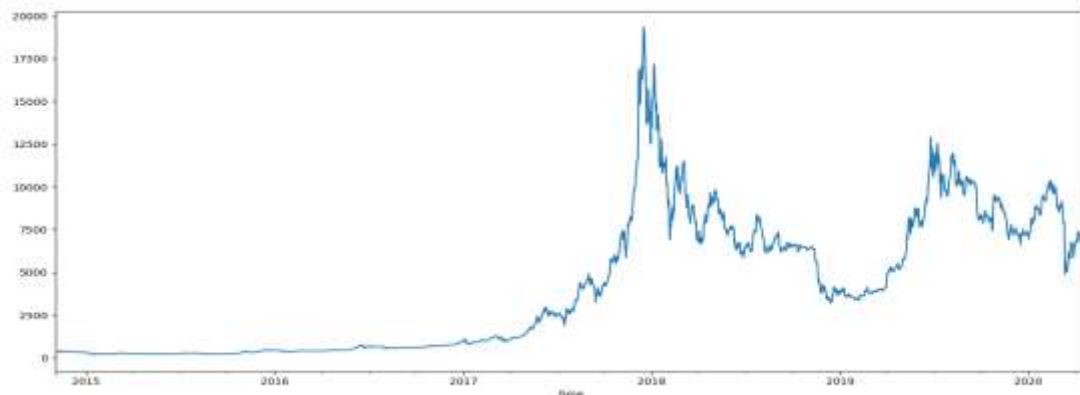


Fig 2: bit coin prediction over years

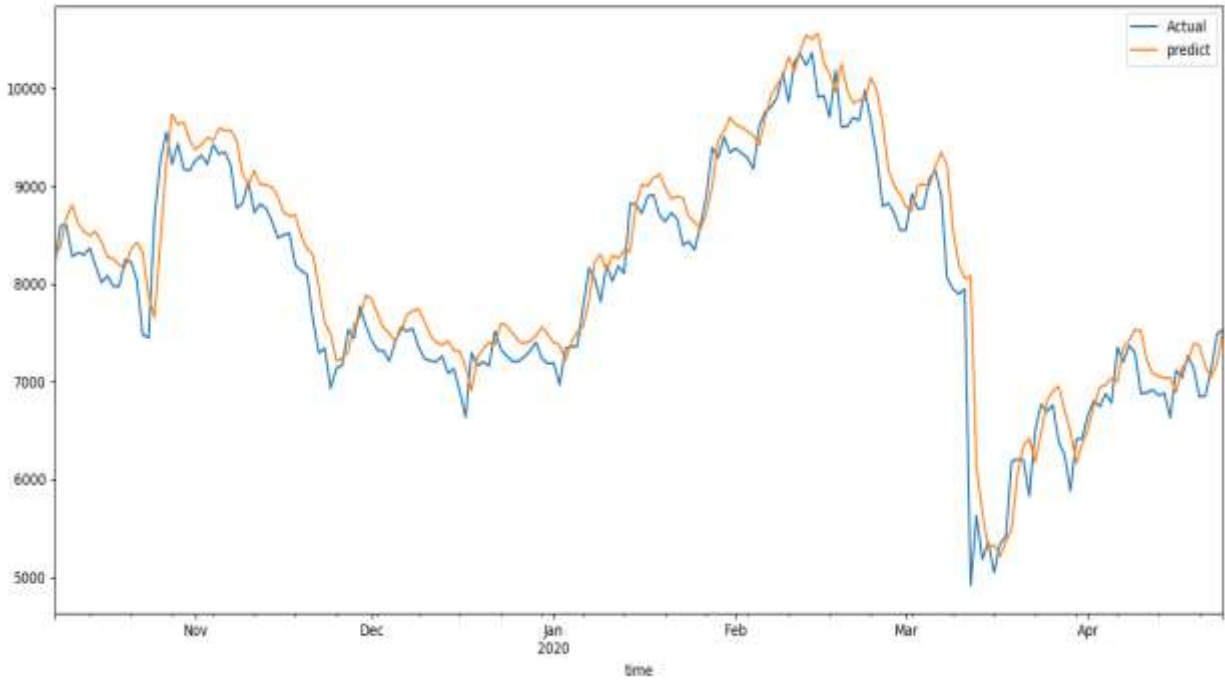


Fig 3: bit coin actual and prediction analysis over years

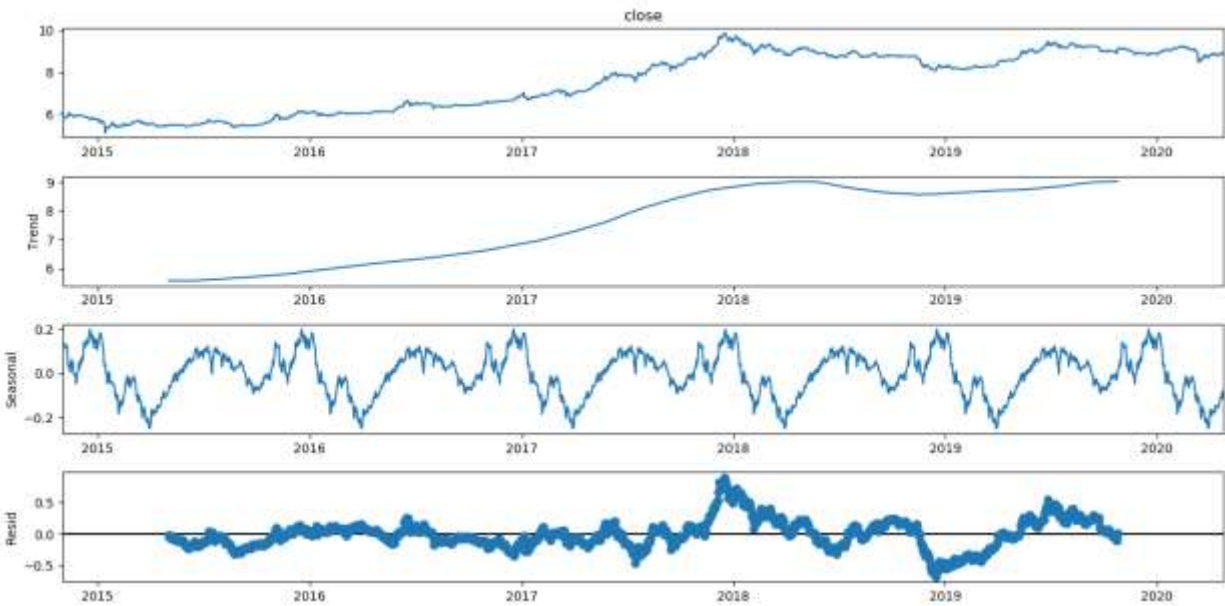


Fig 4: time series analysis of bit coin

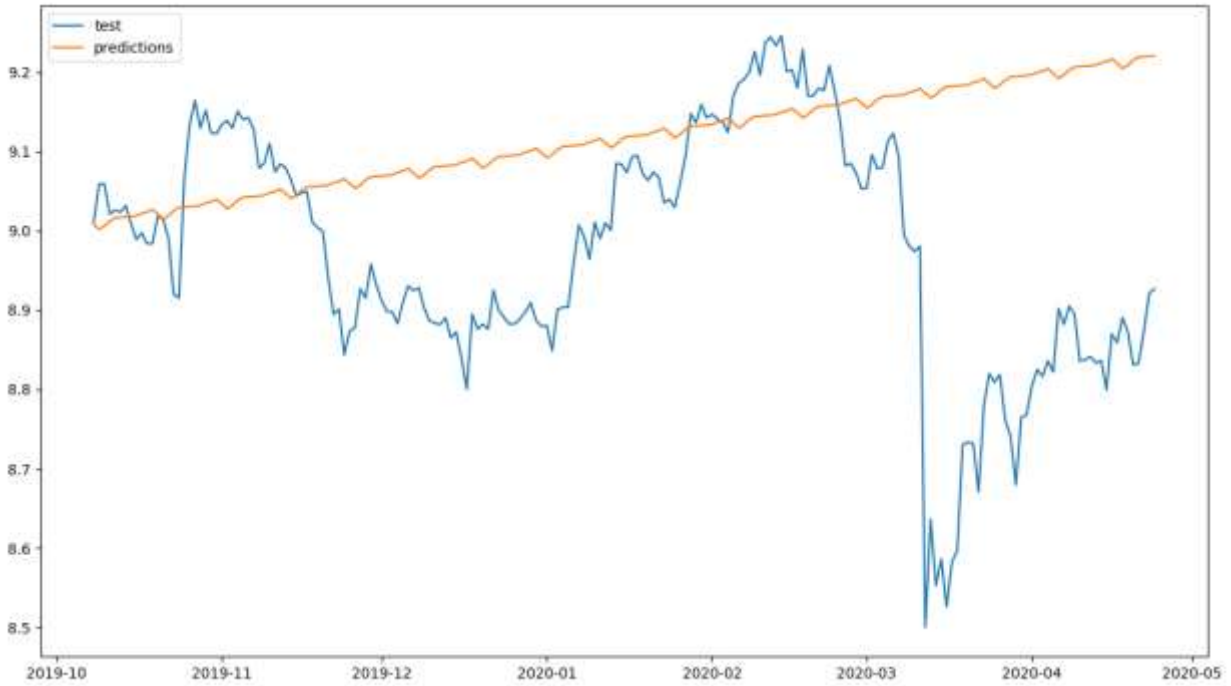


Fig 5: bit coin test and prediction analysis over years

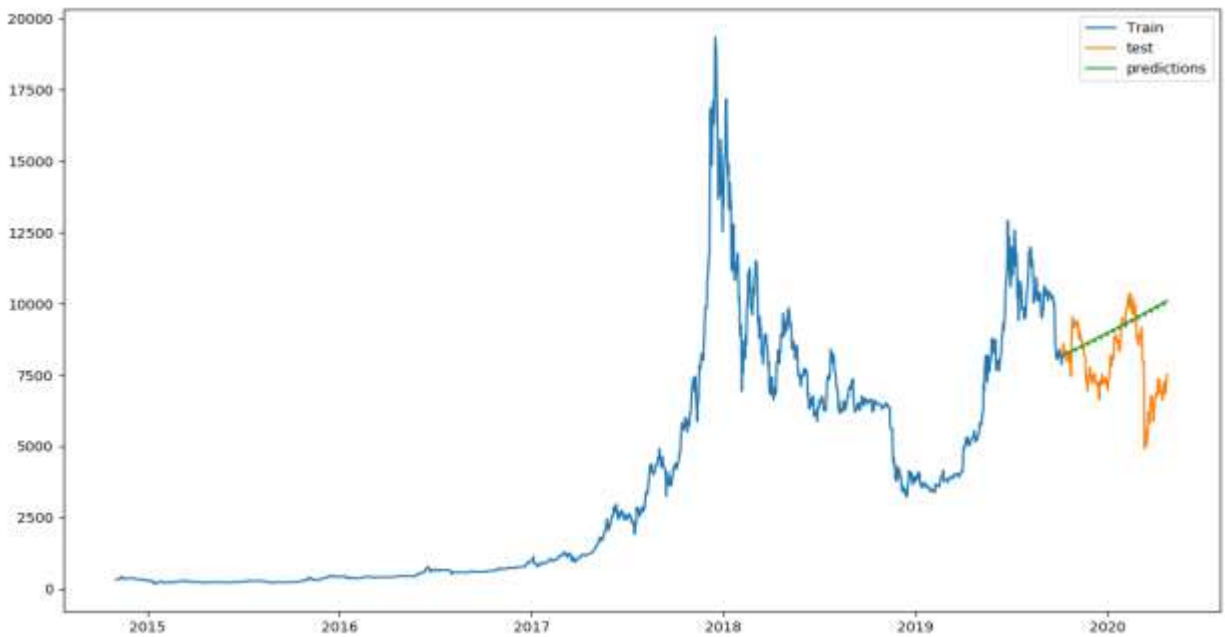


Fig 6: bit coin train, test and prediction analysis over years

The performance benefits gained from the parallelisation of machine learning algorithms on a GPU are evident with a 70.7% performance improvement for training the LSTM model. Looking

at the task from purely a classification perspective it may be possible to achieve better results. One limitation of the research is that the model has not been implemented in a practical or real time setting for predicting into the future as opposed to learning what has already happened. In addition, the ability to predict using streaming data should improve the model. Sliding window validation is an approach not implemented here but this may be explored as future work. One problem that will arise is that the data is inherently shrouded in noise. In terms of the dataset, based on an analysis of the weights of the model the difficulty and hash rate variables could be considered for pruning. Machine learning models require a significant amount of data to learn effectively. If the granularity of data was changed to per minute this would provide 512,640 data points in a year. Data of this nature is not available for the past but is currently being gathered from CoinDesk on a daily basis for future use. Finally, parallelisation of algorithms is not limited to GPU devices. Field Programmable Gate Arrays (FPGA) are an interesting alternative to GPU devices in terms of parallelisation and machine learning models have been shown to perform better on FPGA than on a GPU.

VI. CONCLUSION AND FUTURE WORK

Machine learning models such as the SVR and LSTM are evidently effective for Bitcoin prediction with the LSTM more capable for recognising longer-term dependencies. However, a high variance task of this nature makes it difficult to transpire this into impressive validation results. As a result it remains a difficult task. There is a fine line between overfitting a model and preventing it from learning sufficiently. Dropout is a valuable feature to assist in improving this. However, despite using Bayesian optimisation to optimize the selection of dropout it still couldn't guarantee good validation results. Despite the metrics of sensitivity, specificity and precision indicating good performance, the actual performance of the ARIMA forecast based on error was significantly worse than the neural network models. The LSTM outperformed the SVR marginally, but not significantly. However, the LSTM takes considerably longer to train.

REFERENCES

- [1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.
- [2] M. Briere, K. Oosterlinck, and A. Szafarz, "Virtual currency, tangible ` return: Portfolio diversification with bitcoins," *Tangible Return: Portfolio Diversification with Bitcoins* (September 12, 2013), 2013.
- [3] I. Kaastra and M. Boyd, "Designing a neural network for forecasting financial and economic time series," *Neurocomputing*, vol. 10, no. 3, pp. 215–236, 1996.
- [4] H. White, "Economic prediction using neural networks: The case of ibm daily stock returns," in *Neural Networks, 1988., IEEE International Conference on*. IEEE, 1988, pp. 451–458.
- [5] C. Chatfield and M. Yar, "Holt-winters forecasting: some practical issues," *The Statistician*, pp. 129–140, 1988.
- [6] B. Scott, "Bitcoin academic paper database," *suitpossum blog*, 2016.

- [7] M. D. Rechenhth, “Machine-learning classification techniques for the analysis and prediction of high-frequency stock direction,” 2014.
- [8] D. Shah and K. Zhang, “Bayesian regression and bitcoin,” in *Communication, Control, and Computing (Allerton)*, 2014 52nd Annual Allerton Conference on. IEEE, 2014, pp. 409–414.
- [9] G. H. Chen, S. Nikolov, and D. Shah, “A latent source model for nonparametric time series classification,” in *Advances in Neural Information Processing Systems*, 2013, pp. 1088–1096.
- [10] I. Georgoula, D. Pournarakis, C. Bilanakos, D. N. Sotiropoulos, and G. M. Giaglis, “Using time-series and sentiment analysis to detect the determinants of bitcoin prices,” Available at SSRN 2607167, 2015.
- [11] M. Matta, I. Lunesu, and M. Marchesi, “Bitcoin spread prediction using social and web search media,” *Proceedings of DeCAT*, 2015.
- [12] —, “The predictor impact of web search media on bitcoin trading volumes.”
- [13] B. Gu, P. Konana, A. Liu, B. Rajagopalan, and J. Ghosh, “Identifying information in stock message boards and its implications for stock market efficiency,” in *Workshop on Information Systems and Economics*, Los Angeles, CA, 2006.
- [14] A. Greaves and B. Au, “Using the bitcoin transaction graph to predict the price of bitcoin,” 2015.
- [15] I. Madan, S. Saluja, and A. Zhao, “Automated bitcoin trading via machine learning algorithms,” 2015.
- [16] R. Delfin Vidal, “The fractal nature of bitcoin: Evidence from wavelet power spectra,” *The Fractal Nature of Bitcoin: Evidence from Wavelet Power Spectra (December 4, 2014)*, 2014.
- [17] L. Kristoufek, “What are the main drivers of the bitcoin price? evidence from wavelet coherence analysis,” *PloS one*, vol. 10, no. 4, p. e0123923, 2015.
- [18] Y. Yoon and G. Swales, “Predicting stock price performance: A neural network approach,” in *System Sciences*, 1991. *Proceedings of the Twenty-Fourth Annual Hawaii International Conference on*, vol. 4. IEEE, 1991, pp. 156–162.
- [19] T. Koskela, M. Lehtokangas, J. Saarinen, and K. Kaski, “Time series prediction with multilayer perceptron, fir and elman neural networks,” in *Proceedings of the World Congress on Neural Networks*. Citeseer, 1996, pp. 491–496.
- [20] C. L. Giles, S. Lawrence, and A. C. Tsoi, “Noisy time series prediction using recurrent neural networks and grammatical inference,” *Machine learning*, vol. 44, no. 1-2, pp. 161–183, 2001.