Time Series Analysis of Blockchain-Based Cryptocurrency Price Changes

Jacques Fleischer
jacques.fleischer002@mymdc.net
Miami Dade College
Miami, FL, USA

Gregor von Laszewski
laszewski@gmail.com
University of Virginia
Charlottesville, VA, USA

Carlos Theran
carlos.theran@famu.edu
Florida A&M University
Tallahassee, FL, USA

Yohn Jairo Parra Bautista
yohn.parrabautista@famu.edu
Florida A&M University
Tallahassee, FL, USA

ABSTRACT

In this paper we apply neural networks and Artificial Intelligence (AI) to historical records of high-risk cryptocurrency coins to train a prediction model that guesses their price. This paper's code contains Jupyter notebooks, one of which outputs a timeseries graph of any cryptocurrency price once a CSV file of the historical data is inputted into the program. Another Jupyter notebook trains an LSTM, or a long short-term memory model, to predict a cryptocurrency's closing price. The LSTM is fed the close price, which is the price that the currency has at the end of the day, so it can learn from those values. The notebook creates two sets: a training set and a test set to assess the accuracy of the results.

The data is then normalized using manual min-max scaling so that the model does not experience any bias; this also enhances the performance of the model. Then, the model is trained using three layers— an LSTM, dropout, and dense layer—minimizing the loss through 50 epochs of training; from this training, a recurrent neural network (RNN) is produced and fitted to the training set. Additionally, a graph of the loss over each epoch is produced, with the loss minimizing over time. Finally, the notebook plots a line graph of the actual currency price in red and the predicted price in blue. The process is then repeated for several more cryptocurrencies to compare prediction models. The parameters for the LSTM, such as number of epochs and batch size, are tweaked to try and minimize the root mean square error.

CCS CONCEPTS

• **Applied computing** → **E-commerce infrastructure**; **Forecasting**.

KEYWORDS

blockchain, time series, finance

1 INTRODUCTION

Blockchain is *an open, distributed ledger* which records transactions of cryptocurrency. Systems in blockchain are decentralized, which means that these transactions are shared and distributed among all participants on the blockchain for maximum accountability. Furthermore, this new blockchain technology is becoming an increasingly popular alternative to mainstream transactions through traditional banks [11]. These transactions utilize blockchain-based *cryptocurrency*, which is a popular investment of today's age, particularly in Bitcoin. However, the U.S. Securities and Exchange Commission warns that high-risk accompanies these investments [15].

Artificial Intelligence (AI) can be used to predict the prices' behavior to avoid cryptocurrency coins' severe volatility that can scare away possible investors [9]. AI and blockchain technology make an ideal partnership in data science; the insights generated from the former and the secure environment ensured by the latter create a goldmine for valuable information. For example, an up-and-coming innovation is the automatic trading of *digital investment assets* by AI, which will hugely outperform trading conducted by humans [16]. This innovation would not be possible without the construction of a program which can pinpoint the most ideal time to buy and sell. Similarly, AI is applied in this experiment to predict the future price of cryptocurrencies on a number of different blockchains, including the Electro-Optical System and Ethereum.

Long short-term memory (LSTM) is a neural network (form of AI) which ingests information and processes data using a *gradient-based learning algorithm* [10]. This creates an algorithm that improves with additional parameters; the algorithm *learns* as it ingests. LSTM neural networks will be employed to analyze pre-existing price data so that the model can attempt to generate the future price in varying timetables, such as ten days, several months, or a year from the last date. This innovation could provide as a boon for insights into investments with potentially great returns; it could also contribute to a positive cycle of attracting investors to a coin, which results in a price increase, which repeats. The main objective is to provide insights for investors on an up-and-coming product: cryptocurrency.

2 DATASETS

This paper utilizes yfinance, a Python module which downloads the historical prices of a cryptocurrency from the first day of its

inception to whichever day the program is executed. For example, the Yahoo Finance page for EOS-USD is the source for Figure 1 [3]. Figure 1 shows the historical data on a line graph when the program receives EOS-USD as an input.

Figure 1: Line graph of EOS price from 9 November 2017 to 13 January 2022. Generated using yfinance-lstm.ipynb [8] located in project/code, utilizing price data from Yahoo Finance [3].
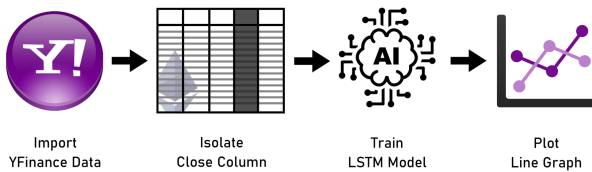
3 ARCHITECTURE

Figure 2: The process of producing LSTM timeseries based on cryptocurrency price.

This program undergoes the four main phases outlined in Figure 2, which are: retrieving data from Yahoo Finance [3], isolating the Close prices (the price the cryptocurrency has at the end of each day), training the LSTM to predict Close prices, and plotting the prediction model, respectively.

4 IMPLEMENTATION

Initially, this program was meant to scrape prices using the BeautifulSoup Python module; however, slight changes in a financial page's website caused the code to break. Alternatively, Kaggle offered historical datasets of cryptocurrency, but they were not up to date. Thus, the final method of retrieving data is from Yahoo Finance through the yfinance Python module, which returns the coins' price from the day to its inception to the present day.

The code is inspired from Towards Data Science articles by Serafeim Loukas [12] and Viraf [14], who explore using LSTM to predict stock timeseries. This program contains adjustments and changes to their code so that cryptocurrency is analyzed instead. We opt to use LSTM (long short-term memory) to predict the price because it has a memory capacity, which is ideal for a timeseries data set analysis such as cryptocurrency price over time. LSTM can remember historical patterns and use them to inform further predictions; it can also selectively choose which datapoints to use and which to disregard for the model [18]. For example, this experiment's code isolates only the close values to predict them and nothing else.

Firstly, the code asks the user for the ticker of the cryptocurrency that is to be predicted, such as EOS-USD or BTCUSD. A complete list of acceptable inputs is under the Symbol column at the Yahoo Finance list of cryptocurrencies [2] but theoretically, the program should be able to analyze traditional stocks as well.

Then, the program downloads the historical data for the corresponding coin through the yfinance Python module [1]. The data must go through normalization for simplicity and optimization of the model. Next, the Close data (the price that the currency has at the end of the day, everyday since the coin's inception) is split into two sets: a training set and a test set, which are further split into their own respective x and y sets to guide the model through training.

The training model is run through a layer of long short-term memory, as well as a dropout layer to prevent overfitting and a dense layer to give the model a memory capacity. Figure 3 showcases the setup of the LSTM layer.

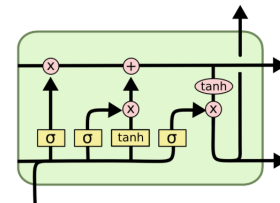The entire program which performs all of the aforementioned steps can be found on GitHub [8].

Figure 3: Visual depiction of one layer of long short-term memory [13].

Figures 4 through 7 use the EOS-USD data set from November 9th, 2017 to January 13th, 2022; furthermore, these figures were all produced within the yfinance-lstm Jupyter Notebook [8]. Within the trained model, only the last 200 days are predicted so that the model can analyze the preexisting data prior to the 200 days for the sake of training.

After training through 50 epochs, the program generated Figure 4, a line graph of the prediction model.

During training, the number of epochs can affect the model loss. According to Figures 6 and 7, the loss starts to minimize around the 25th epoch of training. The greater the number of epochs, the
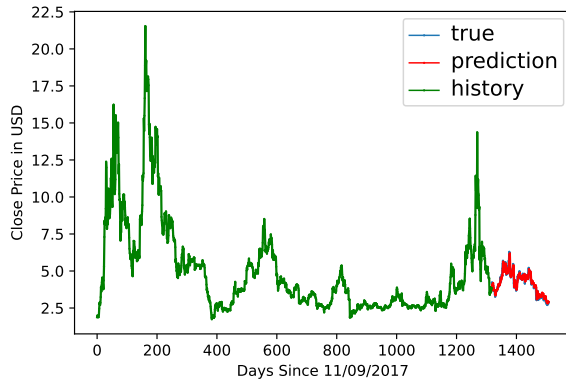
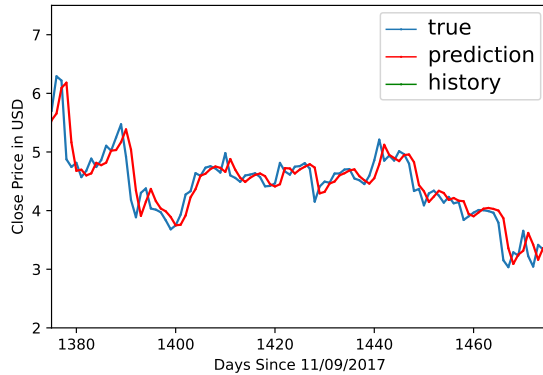**Figure 4: EOS-USD price overlayed with the latest 200 days predicted by LSTM.**

Figure 5: Zoomed-in graph (same as Figure 4 but scaled x and y-axis for readability.

Figure 6: Line graph of model loss over the number of epochs the prediction model completed using EOS-USD data set.

Table 1: Number of epochs compared with Root Mean Squared Error rounded to the nearest thousandth; all tests were run with EOS-USD as input.

Epochs	Root Mean Squared Error
5	0.523 USD
15	0.286 USD
25	0.260 USD
50	0.235 USD
100	0.229 USD

the other coins, likely because most of the training period encompasses a period of relative inactivity (no high changes in price).

5 BENCHMARK

The benchmark is run within yfinance-lstm.ipynb located in project/code [8]. The program ran on a 64-bit Windows 10 Home Edition (21H1) computer with a Ryzen 5 3600 processor (3.6 GHz). It also has dual-channel 16 GB RAM clocked at 3200 MHz and a GTX 1660 Ventus XS OC graphics card. Table 2 lists these specifications as well as the allocated computer memory during runtime and module versions. Table 3 shows that the amount of time it takes to train the 50 epochs for the LSTM is around 15 seconds, while the entire program execution takes around 16 seconds. A StopWatch module was used from the package cloudmesh-common [17] to precisely measure the training time.

In Table 3, the time column reports the length of the program phase in seconds. Training time and prediction time do not perfectly add up to overall time because the time it took to split data into train and test sets is not part of the training or prediction phases. Furthermore, the start times are similar because the entire program's cells were run consecutively.

6 CONCLUSION

At first glance, the results look promising as the predictions have minimal deviation from the true values (as seen in Figure 5). However, upon closer look, the values lag by one day, which is a sign

sharper and more accurate the prediction becomes, but it does not vastly improve after around the 25th epoch.

The number of training epochs can affect the Root Mean Squared Error of the model, which details how close the prediction line is to the real, historical Close prices in United States Dollars (USD). As demonstrated in Table 1, more epochs lessens the Root Mean Squared Error (but the change becomes negligible after 25 epochs).

Figure 7 also shows the impact that epochs have on accuracy. Figure 7 contains two lines: a blue line for the actual price of the EOS coin, and a red line for the model's prediction of the price. As the number of epochs increases, the prediction becomes more and more accurate to the actual price that the cryptocoin was valued at on the market. In Figure 7, the green "history" line is not shown because the graph is zoomed in to the later prediction phase, where the historical price data becomes the blue line instead of green.

Lastly, cryptocurrencies other than EOS such as Dogecoin, Ethereum, and Bitcoin can be analyzed as well. Figure 8 demonstrates the prediction models generated for these cryptocurrencies. Dogecoin presents a model with predictions that are more widely offset than
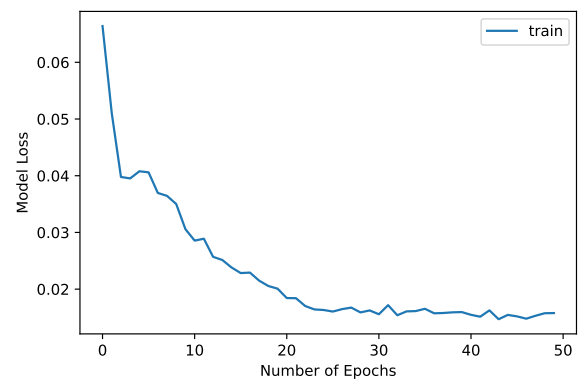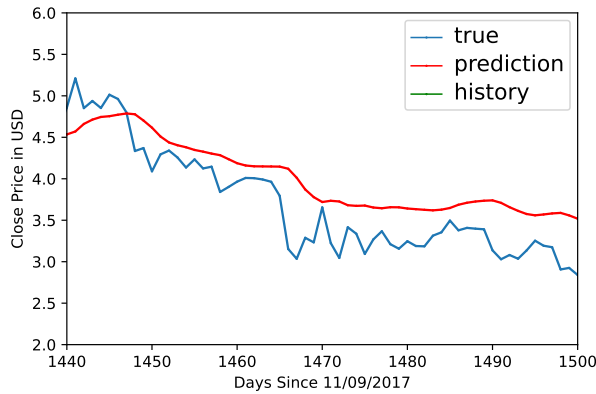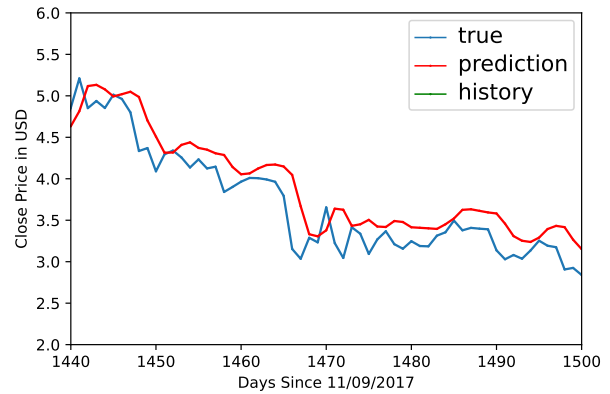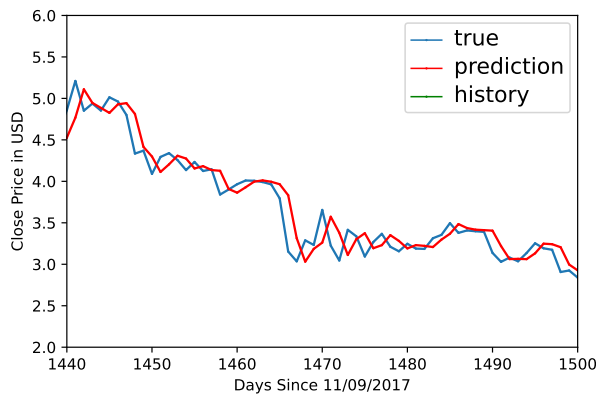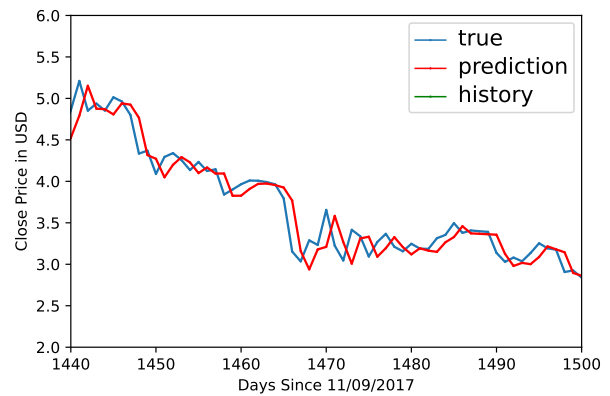
(a) 5 Epochs

(b) 25 Epochs

(c) 50 Epochs

(d) 100 Epochs

Figure 7: Effect of EOS-USD prediction model based on the number of epochs completed.

that they are only viewing the previous day and mimicking those values. Furthermore, the model cannot go several days or years into the future because there is no data to run on, such as opening price or volume. The experiment is further confounded by the nature of stock prices: they follow random walk theory, which means that the nature in which they move follows a random walk: the changes in price do not necessarily happen as a result of previous changes. Thus, this nature of stocks contradicts the very architecture of this experiment because long short-term memory assumes that the values have an effect on one another.

For future research, a program can scrape tweets from influencers' Twitter pages so that a model can guess whether public discussion of a cryptocurrency is favorable or unfavorable (and whether the price will increase as a result).

ACKNOWLEDGMENTS

REFERENCES

[1] Ran Aroussi. 2019. Reliably download historical market data from with Python. https://aroussi.com/post/python-yahoo-finance
[2] Yahoo Finance. [n.d.]. Cryptocurrencies. https://finance.yahoo.com/cryptocurrencies
[3] Yahoo Finance. [n.d.]. EOS USD (EOS-USD). https://finance.yahoo.com/quote/EOS-USD/history?p=EOS-USD
[4] Jacques Fleischer. 2021. Poster of Time Series Analysis of Blockchain-Based Cryptocurrency Price Changes. GitHub. https://github.com/cybertraining-dsc/su21-reu-361/blob/main/project/Poster.pptx?raw=true.
[5] Jacques Fleischer. 2021. Presentation of Time Series Analysis of Blockchain-Based Cryptocurrency Price Changes. GitHub. https://github.com/cybertraining-dsc/su21-reu-361/blob/main/project/Slideshow.pptx?raw=true.
[6] Jacques Fleischer. 2021. README.md Install Documentation. GitHub. https://github.com/cybertraining-dsc/su21-reu-361/blob/main/project/code/README.md

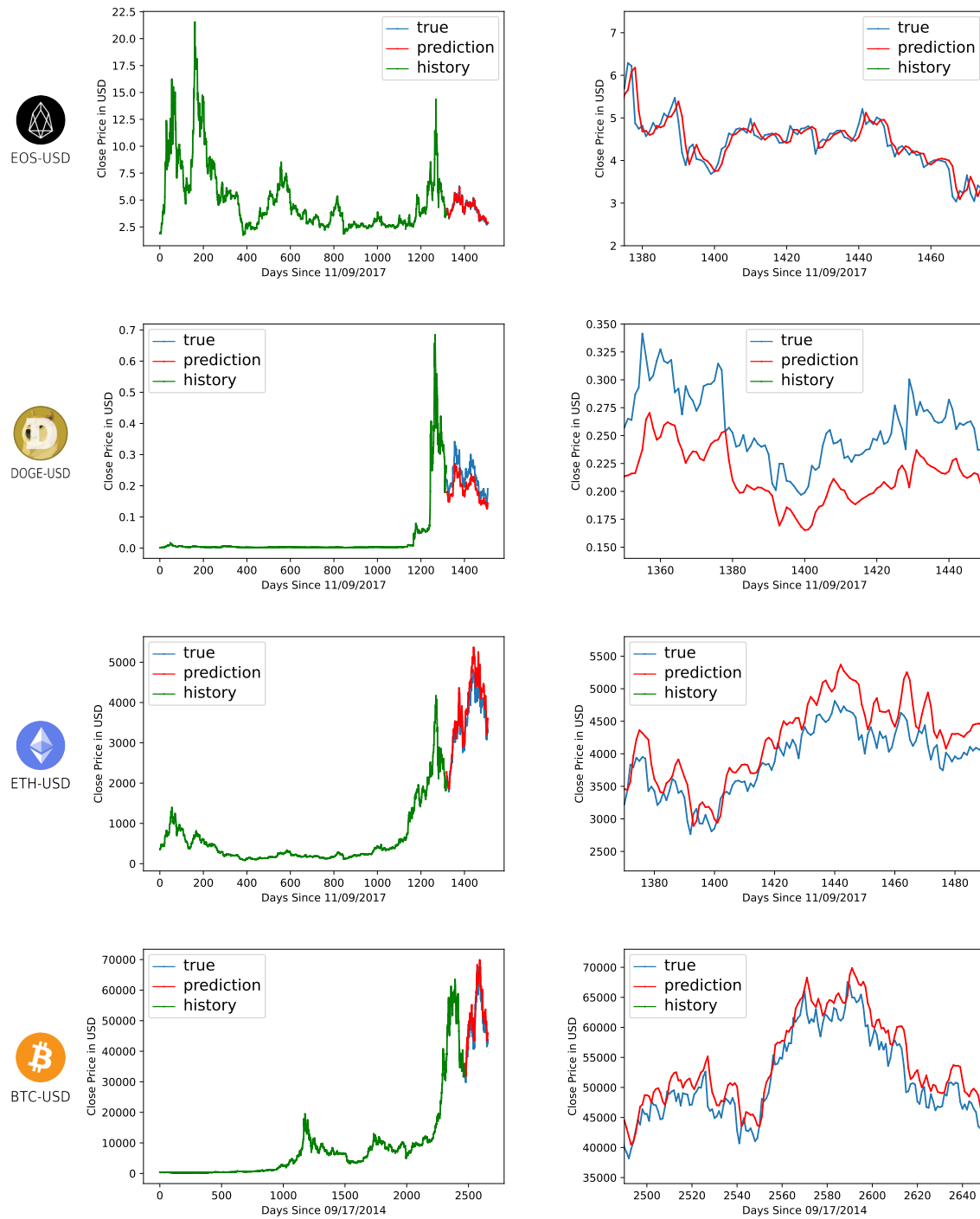**Figure 8: EOS, Dogecoin, Ethereum, and Bitcoin prediction models.**

Table 2: Benchmark details including the specifications and status of the computer at the time of program execution.

Attribute	Value
cpu cores	6
cpu threads	12
cpu frequency	3600.0 MHz
mem.available	7.1 GiB
mem.percent	55.3 %
mem.total	16.0 GiB
mem.used	8.8 GiB
python	3.9.5 (tags/v3.9.5:0a7dcbd, May 3 2021, 17:27:52) [MSC v.1928 64 bit (AMD64)]
python.pip	21.1.3
python.version	3.9.5
uname.processor	AMD64 Family 23 Model 113 Stepping 0, AuthenticAMD
uname.system	Windows
uname.version	10.0.19043

Table 3: Benchmark output which reports the execution time of overall program, training phase, and prediction phase.

Name	Time	Start	OS Version
Overall time	16.589 s	2021-07-26 18:39:57	Windows 10.0.19043, SP0
Training time	15.186 s	2021-07-26 18:39:58	Windows 10.0.19043, SP0
Prediction time	0.227 s	2021-07-26 18:40:13	Windows 10.0.19043, SP0

A ADDITIONAL MATERIAL

The following additional material is available:

Online Description [7]
Install documentation [6]
Python Notebook yfinance-lstm.ipynb [8]
Presentations Presentations of this work were given at the 2021 FAMU-FGLSAMP Data Science and AI Research Experience for Undergraduates Presentation [5] and as poster in the Miami Dade College School of Science 10th Annual STEM Research Symposium 2021 Poster [4].

[7] Jacques Fleischer. 2021. Time Series Analysis of Blockchain-Based Cryptocurrency Price Changes. GitHub. https://github.com/cybertraining-dsc/su21-reu-361/blob/main/project/index.md
[8] Jacques Fleischer. 2021. yfinance-lstm.ipynb Jupyter Notebook. GitHub. https://github.com/cybertraining-dsc/su21-reu-361/blob/main/project/code/yfinance-lstm.ipynb
[9] Jeremy Swinfen Green. 2018. Understanding cryptocurrency market fluctuations. https://www.telegraph.co.uk/business/business-reporter/cryptocurrency-market-fluctuations/
[10] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. Technical report. https://www.bioinf.jku.at/publications/older/2604.pdf
[11] Marco Iansiti and Karim R. Lakhani. 2017. The Truth About Blockchain. https://hbr.org/2017/01/the-truth-about-blockchain
[12] Serafeim Loukas. 2020. Time-Series Forecasting: Predicting Stock Prices Using An LSTM Model. Blog. https://towardsdatascience.com/lstm-time-series-forecasting-predicting-stock-prices-using-an-lstm-model-6223e9644a2f
[13] Christopher Olah. 2015. Understanding LSTM Networks. GitHub. https://colah.github.io/posts/2015-08-Understanding-LSTMs/
[14] Viraf Patrawala. 2020. How (NOT) To Predict Stock Prices With LSTMs. Blog. https://towardsdatascience.com/how-not-to-predict-stock-prices-with-lstms-a51f564ccbca
[15] Lori Schock. [n.d.]. Thinking About Buying the Latest New Cryptocurrency or Token? https://www.investor.gov/additional-resources/spotlight/directors-take/thinking-about-buying-latest-new-cryptocurrency-or
[16] Raj Shroff. 2020. When Blockchain Meets Artificial Intelligence. Blog. https://medium.com/swlh/when-blockchain-meets-artificial-intelligence-e448968d0482
[17] Gregor von Laszewski. [n.d.]. Cloudmesh StopWatch and Benchmark from the Cloudmesh Common Library. GitHub. https://github.com/cloudmesh/cloudmesh-common
[18] Derk Zomer. 2020. Using machine learning to predict future bitcoin prices. Blog. https://towardsdatascience.com/using-machine-learning-to-predict-future-bitcoin-prices-6637e7bfa58f