Featherstone, Jack Bartholomew

**A Final Year Research Project On: Predicting Price Fluctuations of Cryptocurrencies Using a Temporal Convolutional Network**

**Project Student:**

***Jack Featherstone***

**Project Supervisor: *Andrea Visentin***

**Presented To**

**The School of Computer Science and Information Technology.**

**University College Cork**

I believe cryptocurrency fluctuation prediction is a conversation that can be shared among various disciplines, including Computer Science, Mathematics, Data Science and Economics.

It is no secret why price forecasting for cryptocurrency has become a trending research topic globally.

Despite the various factors that contribute to cryptocurrencies volatile behaviour, many machine learning and deep learning networks have been used to predict cryptocurrency prices with great accuracy.

In this project I used a Temporal Convolutional Network (TCN) algorithm to predict the prices of three types of cryptocurrencies, namely Bitcoin (BTC), Ethereum (ETH) and Ripple (XRP).

I used time-series data from 17/08/2017 up until 01/01/2022.

\*results\*

The main goal behind this algorithm was to achieve a reliable dependable model that investors can rely on, based on past cryptocurrency prices.

Contents

[Pulling The Data 2](#_Toc98164295)

[Data Pre-processing 2](#_Toc98164296)

[Picking the Baseline Model 2](#_Toc98164297)

[Trouble Replicating the Baseline’s Model 3](#_Toc98164298)

[Google Cloud 3](#_Toc98164299)

[My Instance 4](#_Toc98164300)

[Baselines Results on my Cryptocurrencies 4](#_Toc98164301)

[Adjusting Hyper-parameters 5](#_Toc98164302)

[Evaluation Metrics 6](#_Toc98164303)

[TCN 6](#_Toc98164304)

[My Github : https://github.com/jackyboy009/PythonCryptoPrediction/tree/master 7](#_Toc98164305)

**Introduction**

Featherstone, Jack Bartholomew

**A Final Year Research Project On: Predicting Price Fluctuations of Cryptocurrencies Using a Temporal Convolutional Network**

**WHERE MY INTEREST STEMMED FROM**

I was introduced to cryptocurrency from my colleagues and friends raving about this get rich quick scheme, so I decided to invest. When I didn’t become a multi-millionaire in one day, I panicked and sold back, I didn’t fully understand the process and technology.

After this escapade I was relatively captured on how people did this for a living and how they decided at what time to invest and what time to sell.

**CRYPTO’S RAISE TO FAME**

Since April 2011, when Bitcoin first surpassed one dollar, the term crypto has become progressively recognized as something that is here to stay, rather than some convoluted pyramid-scheme.

At the time of writing this 10/03/2022, the global crypto market capitalization is 1.83 trillion dollars with 18,000 crypto currencies available. – from Investopedia

Binance is one of the largest crypto exchanges in the world, figure 1 shows the increasing rate of users per year.

Lark Davis presented the figure below which depicts the whole concept of crypto having the highest adoption rate of any technology in human history. It is being compared with the initial growth of the internet, and we all know how that story ended.

Table

Description automatically generated

**Figure 1.** Binance users per year

Chart, line chart

Description automatically generated

**Figure 2.** Adoption rate of the term ‘crypto’

**CRYPTO VS STOCKS**

Crypto is a piece of data that is used as a medium of exchange without the need of a third-party intervening (bank). Its goal is to be as commonly accepted as cash or credit.

There is this common misnomer that crypto is just like a stock, this is just not the case. When we buy a stock offered by a particular company, we own a percentage in that company (assets / profit).

When we buy crypto, we are given a certain amount of that digital currency in which we can do as we please. We only own the rights of a particular amount of a digital currency.

Although Stocks and Crypto are fundamentally different, the way in which they are treated are quite typically the same. As of 2022 most day-day companies still do not accept crypto as an acceptable form of credit. As a result, the primary role crypto plays are a store of value in which you can hold onto or sell. This is the same ideology as stock market operates on only rather than store of value, you own a store of ownership.

It is no secret that Stock markets are rather unpredictable and are affected by many factors causing the high volatility in the market. The exact same can be said about cryptocurrency if not more factors that contribute to the volatile behaviour, one rather comical example of this would be the ‘Musk Effect’ a term coined from when Elon Musk’s tweets had a direct and substantial effect on the price of certain crypto’s. With factors such as media hype lending itself as a cause to why crypto produces unstable behaviour, as well as many other factors, predicting something in this unstable market will be extremely difficult.

**IMPORTANCE**

Before crypto, Investors were always on the search for tools and techniques that would increase profit and reduce risk within the stock market. Buying a certain stock involves risk,

the aim of the investor is to keep this risk as low as possible while maximizing profits. So naturally, any instrument that could minimise the risk would be valued highly.

Since the virtual explosion of crypto as outlined above, the exact same can now be said about cryptocurrency price prediction. Cryptocurrency price prediction has become a trending research topic globally and it has created a big opportunity for research.

The importance of being able to predict these high valued currencies, cannot be understated.

Students from various majors are constantly working on ways to more accurately predict the upswing or downswing of a particular currency to know when to invest and when not to.

I believe as technology is advancing, this is the opportunity to find the most informative indicators to make better predictions.

**MODEL I USED**

I used a machine learning model for time series forecasting to accurately predict fluctuations of crypto prices.

A time series is a sequence of data points that are listed in order of time. Time-series forecasting is a common technique used in many real-world applications such as weather forecasting and financial market prediction. It uses the continuous data for a particular period in time, to predict the result for the next period in time (e.g., using the past 3 years of daily temperature in a particular area, to predict next week’s temperature in the same area).

The stock market and now crypto market is a typical area that presents time-series data and many researchers have proposed various models using this data, for various different aims.

Many time-series forecasting algorithms have shown their effectiveness in practice. The most common algorithms are now based on Long-Short Term Memory Networks which are based off Recurrent Neural Networks. One model that was been showing promising results in the Financial Time Series is the Temporal Convolutional Network (TCN).

In this project, I applied a TCN to predict the upswing or downswing of a particular crypto based off its closing price (The price this crypto was during a specific time of the day).

Abstract

\*\*\*

One page summary of the report , including the key results. Standalone page .

For people who have only time to read one page (100 words)

Advert for the rest of the report : after reading this they should know if they are interested or not.

\*\*\*

Target Audience = Assume that people reading have knowledge on the mandatory modules but no optional modules.

Don’t go into too much detail . Don’t assume that the reader has knowledge they don’t .

How ? < 1 page , general outline of the approach to give direction to the thesis

**Introduction**

This project commenced like a lot of deep learning projects do, by collection of the cryptocurrency data!

The cryptocurrency data that I am using in this project is being pulled from Binance’s API. This was a relatively easy process as there is a python library called “python-binance” that helps the user extract cryptocurrency data.

The only problem I encountered was that the API only allowed you to pull at maximum two thousand timesteps, a rather simple solution to this was extracting the data month by month, then merging each individual list together.

I created a function in Python that extracted any information about any cryptocurrency within a specified date range. So that in the case of anything happening to the data, I would not have to go through the monotonous process of collecting all the data again.

# Data Pre-processing

At the time of this project, I was studying an introductory course to Artificial Intelligence, I was relatively familiar with libraries such as “numpy” and “pandas” for the data manipulation.

Saying this, I learned an enormous amount about pre-processing time series data. The most prevalent thing that came to mind is the “Sliding Window” method to model my data. The concept at first seemed quite strange to me, as I was getting the target values from the data that I would ultimately be training on. Although after some clarification it soon became clear to me. The sliding window method was ultimately done before splitting my data into the relevant training, validation and test sets. This way I was saving more data whilst retrieving the most relevant data.

A simple mistake that could be made on standardizing the data is, standardizing the whole data set rather than just the training set. I fell victim to this and only realized rather late into the project.

# Picking the Baseline Model

The first course of action that had to be taken was establishing a baseline model.

After an appropriate amount of reading and enquiring on numerous articles, we decided on

“A Deep Learning-based Cryptocurrency Price Prediction Scheme for Financial Institutions”.

The reasons behind picking this in particular model was that it was recently published - 04/06/2020, it had been moderately cited and its subject matter suited our project description.

As a little time had passed since the publishing of this article, I knew that there would be more cryptocurrency data available to use for training. This could only have been deemed a good thing as the more training data available, the more chance of generalizing the new unseen data. This meant there was the possibility of improving upon the baseline model.

The reasons behind picking a baseline initially was that we could directly compare my findings with the standard model baseline and also get some practice building their models from this article.

A benefit that I found in reading similar projects was that you would learn the accepted standard for data preparation with time series data. For example, a common occurrence across every article I read was the same manipulation to the data (rolling window/ normalizing the data).

Picking a baseline and simulating their model was extremely beneficial to me as I gained experiencing simulating their model. I was able to appreciate that even though my reference came from a published article there could still be some flaws in it, for example, they did not seem to notice that they left some leakage in their model.

# Trouble Replicating the Baseline’s Model

The model proposed in the Baseline is a hybrid model, consisting of two Long Term Short-Term Memory Networks (LSTM) networks and a single Gated Recurrent Unit (GRU) network.

LSTM’s and GRU’s serve relatively the same function (attempting to capture long term temporal patterns). Although GRU’s use two gates inside its cell, rather than three like in a LSTM cell. This modification makes the GRU less computationally expensive as it does not have to carry an additional vector of numbers.

As we have both networks in this hybrid model, the training process for this baseline model seemed to be too much of a challenge for my local machine to handle.

The estimated time it would have taken to run one epoch was nine minutes, as you can imagine if you have quite a large number of epochs, like we did, then this training process seemed like quite a frivolous one. One would be completely deterred from testing out different values for all the hyper-parameters as a result of the time it would take.

# Google Cloud

At this point in the project, I was quite disheartened as it seemed completely unfeasible to run on my local machine, purely because of the time it would take.

The solution? *Creating an instance on Google Cloud.*

This means that a virtual machine was doing all the heavy computation in the cloud.

The process of creating an instance on Google Cloud was rather easy, although saying this, at first, I was not getting the results I had hoped. Rather than getting nine minutes per epoch, I was now getting twelve minutes per epoch!

After a large amount of research and a great deal of help from some of Andrea’s students, the problem was not with Google Cloud, sadly it was a testament to my code.

My training runs were not making the most out of the GPU supplied by the virtual machine.

Increasing the batch size, and also getting rid of the data generators increased the usage of the Graphics Processing Unit (GPU). By calculating the batches in parallel in the GPU memory, this reduced the communication overhead in the CPU. However, it was about finding the balance because I didn’t want the batch sizes too big so that the model could not converge to the best of its ability. The batch size which I ended up deciding on was 512.

# My Instance

The specification of my model are as below:

**GPU**: NVIDIA Tesla T4

**CPU**: 8, 30gb memory

**Pre – Packaged with**: TensorFlow2.8/Keras.CUDA11.3.GPU

**Boot Disk Image:**  Debian GNU/Linux 10 (buster) with 200gb (standard persistent disk).

*The NVIDIA Tesla T4* was picked because it was specifically designed for high performance computing and deep learning training. This GPU believe it or not, was the least powerful GPU on offer. The option was there to upgrade the hardware including the GPU in the virtual machine, although I did not find it necessary as I did not have an extremely complex model, nor an enormous amount of data.

My reasoning behind picking the remaining specification is because this is what was recommended on the deep learning, pre-built instance. This pre-built instance was extremely helpful as I did not have to look for all the necessary software myself as it came pre-loaded with all the necessary packages out of the box.

# Baselines Results on my Cryptocurrencies

Before I could test the model and get the appropriate evaluation metrics, I first had to train the model and plot the training and validation losses.

The baseline model used two dropout layers, so I assumed that without any dropout layers the model would overfit. To my surprise the model did not overfit, it did the opposite, it underfit! (See picture below) Again, I assumed that the reason for this was because we had more datapoints than our baseline did (2000 to our 40,000). It turns out that the reason for this behaviour was because of the vanishing gradient problem.

The reason why the model was underfitting with no regularisation is because the first layers of our network were simply unable to learn anything. As back propagation operated the gradients were getting smaller and smaller until the difference in the updated weights in the early layers were miniscule.

It seems that adding the dropout layers was helping avert the forementioned problem with vanishing gradience andit was helping generalize.

Before Dropout After Dropout

Chart

Description automatically generatedChart, histogram

Description automatically generated

As an experiment I decided to add Relu activation functions to the LSTM and GRU Layers, as Relu returns 1 if the value is greater than 0 , thus as the network increase the gradients do not get smaller. The results of this were similar to the graph above Before Dropout.

# Adjusting Hyper-parameters

The goal was to find the ideal medium between neither underfitting nor overfitting. To achieve the best results, I adjusted various hyper-parameters such as, the size of the dense layers, the batch size, as well as the window size in training the model.

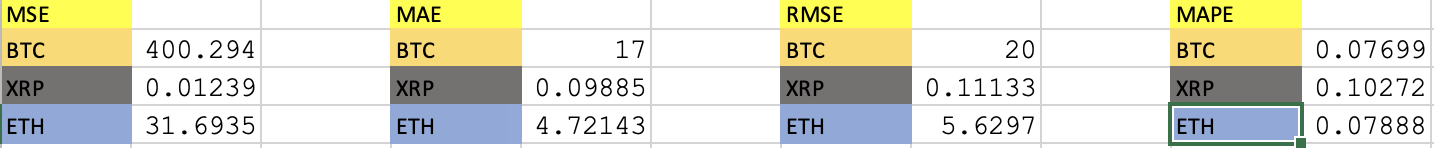
The best results seem to be achieved using a window size of 24 (perhaps because 24 hours is one full day), batch size of 512 (batch size had to be rather large so cloud could utilize its GPU’s) and dense layers of size 128 (enough to capture relevant patterns.)

It was pointed out to me that sometimes we need to predict more time in the future to avail of a certain prediction. The forecast horizon I used was the same as the Baselines, although instead of days like they used, I was using hours. So, we had 1 hour, 3-hour, and 7-hour predictions.

It is important to note, when I was happy with the hyper-parameters I trained on the combination of the training and validation sets and then tested on the test set. This was making avail of all the available data. Testing on the test set was only done after hyper-parameters were fine-tuned on validation and training sets, to avoid any sort of leakage.

# Evaluation Metrics

The evaluation metrics I recorded were the same as those in the original baseline report (MSE, RMSE, MAE, MAPE). These figures were used as a guideline for the results produced by the TCN.

One Hour Prediction 

3 Hour Prediction

Table

Description automatically generated

7 Hour Prediction

Table

Description automatically generated

As the Baseline is using a different crypto currency with scaled data, it is hard to tell if the results are similar in comparison. Nevertheless, these are the figures we will be using as a benchmark.

# TCN

So far, in creating my TCN model I have created the first shell and have made sure the dilations and padding produces the right model description. While studying a network it was hard for me to picture it without knowing exactly how it worked. Most of these networks took a long time to comprehend leading to many sketches, like the figures given below.

# My Github : https://github.com/jackyboy009/PythonCryptoPrediction/tree/master

This is an example of casual convolutions and Dilations within my TCN network

Timeline

Description automatically generated

Timeline

Description automatically generated with medium confidence

The Baseline Model

Diagram

Description automatically generated