Artificial Intelligence: Cryptocurrency Price Predictor

Daniela Beckelhymer, Devon Fulcher, Bobby Garza, Alexander Hoffman

May 2019

1 Abstract

Cryptocurrencies are a novel financial instrument unlike any other asset that has come before them. Being so new, there is very little research into the prediction of their prices as there is for other assests. Producing an effective prediction mechanism could allow for effective trading policies and ultimately revenue. TCNs model the history of a sequence by sliding a receptive field across it. This is unlike recurrent neural networks that model the history of a sequence with recurrent links. TCNs have shown to have greater success than recurrent neural networks in sequence modeling tasks [3]. This is why we used Keras-TCN, a TCN library built in Keras, for price prediction. This uses a one dimensional TCN that can predict future time series data given a history of data. We applied this model to price data for the cryptocurrency Bitcoin and for the commodity gold. With the use of this time series data, we analyze the effects of numerous hyper-parameters on the efficacy of price prediction. We have found that the performance of a model is more dependent on the intersection of many hyper-parameters rather than any single hyper-parameter.

2 Introduction

Our group was primarily interested in being able to predict the price of cryptocurrencies in order to make more intelligent investment decisions based on existing cryptocurrency price data. It is critical for an investor to know the current price of an asset he or she plans to own or sell in the future, but if the price of a asset can be accurately predicted the return on investment for potential users can be optimized for selling or buying. Bitcoin is extremely volatile making price analysis a more difficult task than other with assets. For this reason, our research team has decided to predict price data for Bitcoin as well as less volatile commodities like Gold. We decided to model our prediction with a Temporal Convolutional Neural Network (TCN) because that architecture has been shown to be effective in producing natural sounding human-like

language [1] and more effective than recurrent neural networks in sequence modeling [3]. When this model was applied to Bitcoin and gold price data, we found that the intersection of multiple hyper-parameters has a larger effect on model performance than any one hyper-parameter.

3 Background and Related Work

This project uses time series data in order to perform predictions for future events [2]. Examples of time series data includes weather forecasting, daily reports of items sold, number of traffic accidents per day, etc. In this particular project, we are analyzing observations in changes of price data over the duration of different amounts of time, to see if we can create an artificial intelligence agent that predicts future behavior accurately. Specifically, our group used a TCN to try to predict the price data for different assets.

3.1 Temporal Convolutional Networks

As previously stated, Temporal Convolutional Networks are a type of a CNN that work with respect to time [3].There are two principle parts that build TCNs: having no leakage of data from past to future, and the network producing output that is the exact same length as its input. Additionally, TCNs are usually causal convolutions [3]. Causal convolutions depend solely on past data and not on future input. This is particularly important for our project because the data we are working with is simply price data from the past. There is no way for us to input future price data to predict future price data. Ultimately, we learned that TCNs have outperformed their recurrent counterparts because they exhibit longer memory than recurrent architectures with the same capacity. This was important to understand during our search for different architectures.

3.2 WaveNet

WaveNet is one of the first projects to utilize a convolutional neural network with respect to time. The researchers used text to speech audio data and used a TCN to generate human-language-like sounds that are actually nonexistent. This project is similar to work using GANs with celebrity faces as input to create unique faces as the output. Results of this project proved that the TCN algorithm they created outperformed the current best text-to-speech systems when judged for naturalness [1].

3.3 Bitcoin With Bayesian Neural Networks

Jang and Lee [4] explain the importance of using time series for predicting price data. Rather than using TCNs, this project used a Bayesian Neural Network (BNN) with Bitcoin price data. Bayesian Neural Networks are a type of multi-layered perceptron that has been previously successful with image recognition

and pattern recognition tasks. This made it a potentially beneficial algorithm to use with a time series task. Unlike CNNs that maximize marginal likelihood, BNNs maximizes the value of posterior through an application of the Bayes' theory. Ultimately, this work showed that unlike other benchmark models that fail directional prediction, the BNN model succeeded in relatively accurate direction prediction [4].

4 Domain

The domains used are asset price data, namely Bitcoin and gold. Bitcoin is the most popular in cryptocurrency markets and is a good representation of the high volatility in cryptocurrencies [6]. The 30 day standard deviation for Bitcoin to USD data is 5.4% [8]. This is a high standard deviation when compared to other assets such as gold which is 1.2% [8]. Gold was chosen because it is much more stable than Bitcoin and has a much longer market history. Predicting gold prices is a good way to test how well our implementation can be ported to other asset or even other spaces. Additionally, comparing our model with stable asset, such as gold, to a volatile asset, such as Bitcoin, allows us to better examine the effects of volatility on our model.

5 Approach

At the beginning of this project, our group did research on past prediction projects and looked for base code that we could improve. Our team decided to work upon base code authored by Phillipe Remy on GitHub [7]. The base code includes the method for compiling the TCN. Next, we considered different sources for cryptocurrency data. Blockchain.com and Datahub offered data that most fit our needs. Through partner programming and trial and error, we adapted our code. Our work is distinct from Wavenet [1] in that we applied a TCN to asset price data instead of audio data. Additionally, our work is separate from work with Bitcoin prices and BNNs [4] because TCNs possess a different architecture than BNNs.

5.1 Our Contributions

After figuring that there was a great deal of different combinations for our hyper-parameters, we reorganized our goals. Our team needed to create a way to test different combinations of hyper-parameters quickly and efficiently. Doing so manually was not going to allow us to meet our deadlines. Finally, we decided that we would utilize random search on the input hyper-parameters to determine the best values for them. These hyper-parameters are the kernel size, number of dilations, number of stacks of residual blocks, the use of skip links, and dropout rate. Random search can be more effective than grid search in many data sets [3]. We chose to use random search because it can easily be done asynchronously across all of our computers. Also, random search gives a good

overview of the hyper-parameter space without exhaustively testing many different combinations and random search can be terminated at any time without harming the integrity of the experiment. Additionally, we attained gold price data from Datahub. Because gold is a more stable asset, our team thought it was useful to try to predict its data and compare the losses to Bitcoin. We also performed random search on this data to find optimal hyper-parameters. For each set of uniformly sampled hyperparameters, our model outputted comma separated value documents that contained the loss value and the value of each hyperparameter. We then aggregated this data.

Our final procedure is further discussed thoroughly in Experimental Procedure.

6 Experimental Procedure

Our code continually uniformly samples hyper-parameters from specified ranges. With each sampling of hyper-parameters, our TCN model is trained on either Bitcoin price data or Gold price data. Our Bitcoin price data is from every third day starting on 8/17/10 (the earliest date that we have price information available) to 4/23/19 (the date that we started training our model). We performed 48 different iterations of random search for the Bitcoin data, and each iteration took about an hour to run. We performed 11 iterations for gold price data, which lasted 30 minutes each.

6.1 Kernel Size

Kernel size affects the focus of the model. A larger kernel size means that the model accounts for a greater breadth of the input space per input neuron. We uniformly sampled integers in the range of $\{2, 10\}$ for kernel size.

6.2 Number of Dilations

The number of dilations defines the spread of neurons at each layer of the neural network. The dilations used to take the form of increasing powers of two meaning that each layer has a spread that is a power of two. The length of dilations is how many powers of two are utilized. This length is a uniformly sampled integer in the range $\{3, 9\}$.

6.3 Residual Blocks

A residual block is a set of layers that have a skipped connection across them. The hyper-parameter of the number of residual blocks defines how many of these blocks exist in the network. With a greater number of residual blocks, the network can abate the effects of the vanishing gradient problem. This hyper-parameter value is an integer that is uniformly sampled in the range $\{0, 5\}$.

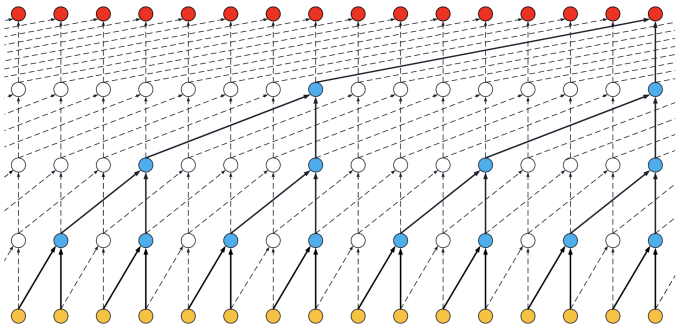

Figure 1: A temporal convolutional network with a kernel size of 2 and dilations 1, 2, 4, 8.

6.4 Skip Links

Skip links is a parameters that defines whether or not a connection will be made from the input to each residual block. This also affects the vanishing gradient problem. The skip links parameter was uniformly sampled in the range {true, false}.

6.5 Dropout Rate

Dropout rate affects how generalized the model is to new data. Dropout is a regularization technique that randomly removes and reinserts neurons in the network so that the network is not overly dependent on too few neurons. Dropout rate determines the rate at which neurons are removed and reinserted. A higher dropout rate means that the model has a greater ability to generalize to new data but also may take longer to train. We uniformly sampled dropout rate in the range {0.01, 0.1}.

7 Gold Prices with our Model

In order to to see if our algorithm was better at predicting a more stable asset, we found yearly data for gold dating back to 1950. After running this data through our algorithm, we found that our loss was significantly less than what it was for when we tested it with Bitcoin data. The average loss was only 7,855.00 compared to 34,781.42 for Bitcoin. This meant it was better at predicting the more stable commodity as we expected. [5]

8 Results

After the 48 iterations for Bitcoin, our team found that the set of hyperparameters with the lowest average loss of 7,158.63 had a kernel of size 2, 5 dilations,

used 0 stacks, used no skip links, and had a dropout rate of 0.015103. The overall average loss for our Bitcoin data was 34,781.42. The median average loss was 27,906.63, and the highest average loss of 101,413.6 had a kernal of 8, 9 dilations, used 0 stacks, used no skip links, and had a dropout rate of 0.032671. After 11 iterations for gold, we found our overall average loss for gold was 7,854.95. The minimum average loss was 4,452.39, the median average loss was 7,042.96 and the maximum was 16,476.2. We did not analyze the hyper-parameters for gold, because we had too few iterations. It is also important to note that because our hyper-parameters do not work independently of one another, the statistical results of each below can only be interpreted in a limited way.

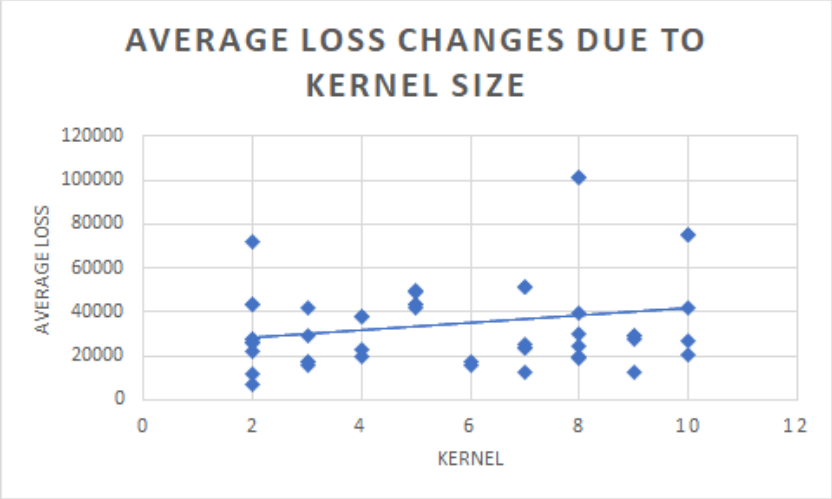

Figure 2: Affect of kernel size

For kernel, we see that typically a lower number results in lower loss. A kernel of size 4 to 7 seems to have less incidences of high loss than larger sizes.

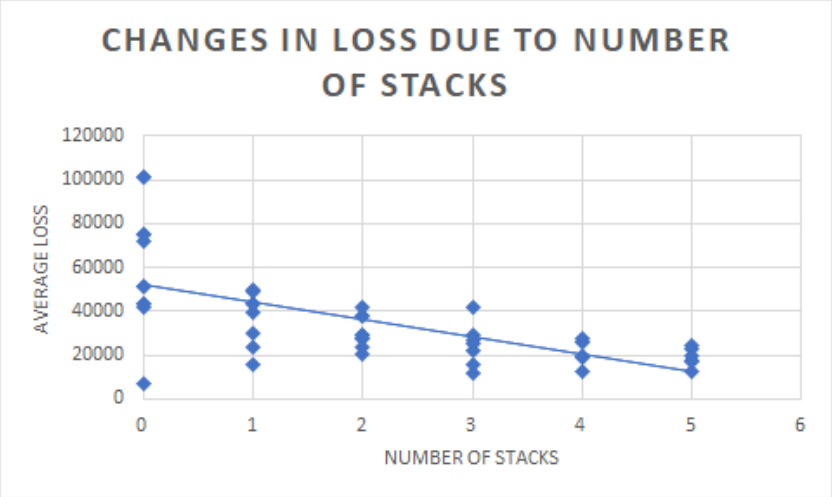

Figure 3: Affect of number of residual blocks

Number of stacks of residual blocks appears to be the most influential hyper-parameter, because it has a stronger slope. As we can see in figure 3, the 0 stacks has the least amount of loss. Then 5 stacks has the next least amounts of loss.

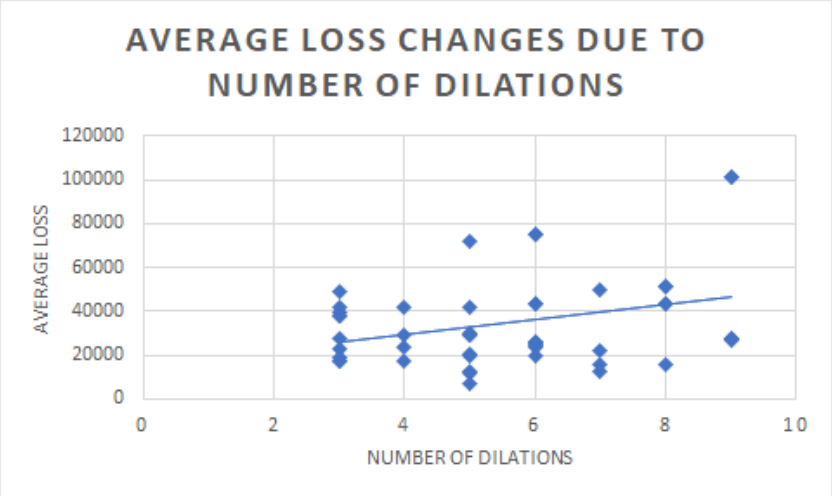

Figure 4: Affect of number of dilations

Number of dilations has clusters of low loss between 4 and 6, some really high loss between 8 and 10, and consistently lower loss between 2 and 4.

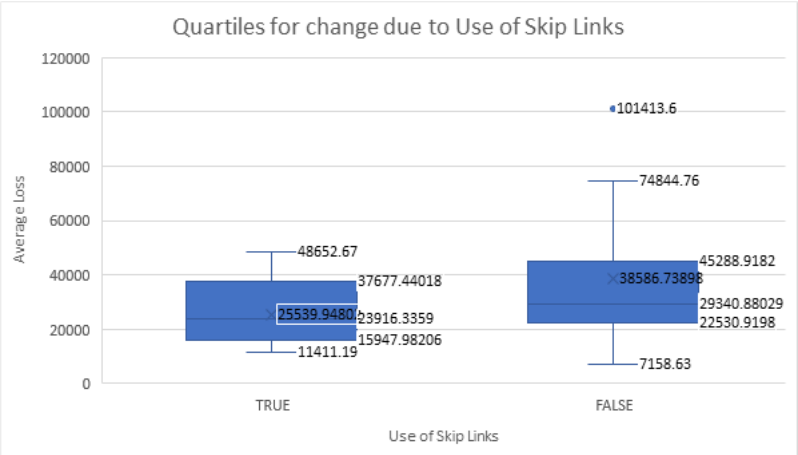

Figure 5: Affect of skip links

In this figure, we used a box plot to show how the use of skip links, whether true or false, affected average loss. The box plot shows descriptive statistics for the data of each. Generally, hyperparameter sets with true for this boolean had lower average losses.

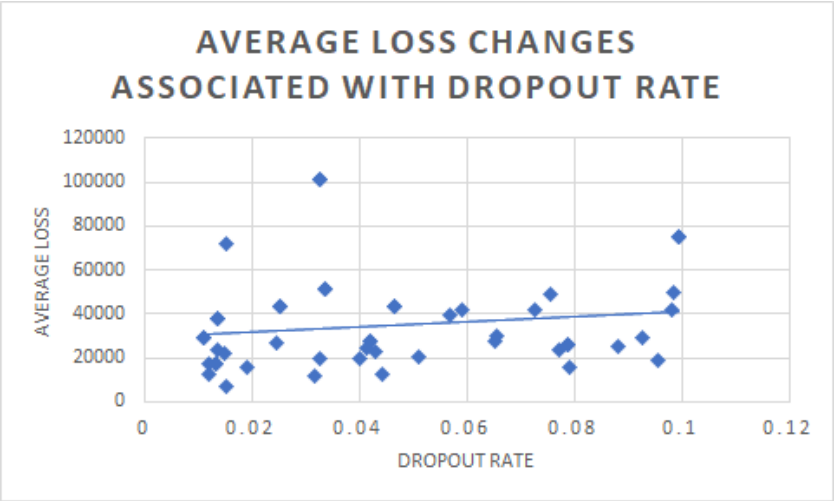

Figure 6: Affect of dropout rate

Dropout rate seems to be positively associated with loss, meaning that the higher the rate the higher the average loss was. However, as seen in the figure there are outliers for extremely high loss that are between 0 and 0.04 for dropout. We can infer that this is due to variability in the sets of hyperparameters where these dropout rates are.

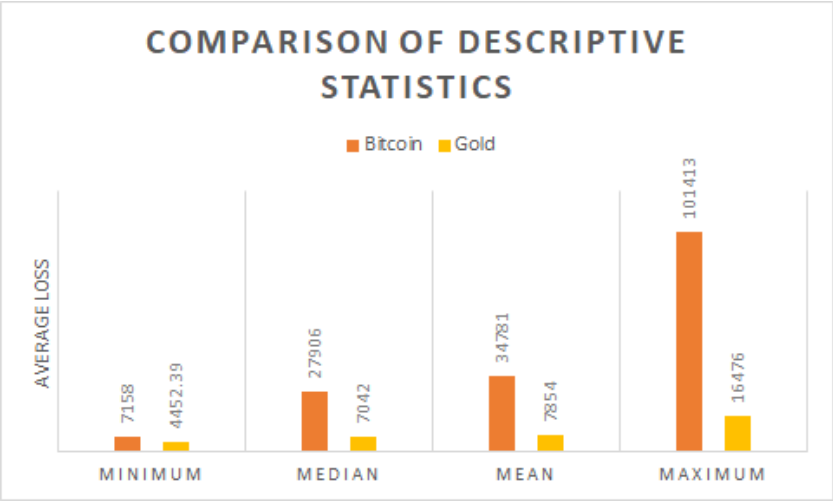

Figure 7: Comparison between Bitcoin and Gold

Finally, when we compare the results for bitcoin and gold we see that gold had significantly lower average loss scores than bitcoin did. Bitcoin seemed to have an exponential growth in its average loss through random search for optimal hyperparameters whereas gold had a more linear increase.

9 Discussion

Throughout this project we have tried different approaches to find a way to predict the future price of Bitcoin. We applied random search to finding hyper-parameters to see if there was a way to find an optimal output. Despite varied approaches our predictions have been far from stellar. This may just be because Bitcoin is too volatile or subject too much speculation. However, we were able to provide data on the efficacy of numerous hyper-parameters. The selection of thoughtful hyper-parameters can increase the chances of creating an effective model but there is little known about the effects that these hyper-parameters have on each other. For instance, one of the most effective models that we created had 0 stacks of residual blocks (see Figure 3). However, a high number of stacks of residual blocks is usually associated with a lower loss and therefore greater predictions. This outlier can be explained by the fact that models are composed of the intersection of multiple hyper-parameters and a single hyper-parameter cannot explain the performance of a model. Furthermore, Bitcoin may be too unstable to accurately predict on at the moment. If it decreases in volatility then it may become more feasible to attempt this again and analyze the results. It would appear that our attempt to predict the price of gold was much more successful, with much lower loss levels when compared with Bitcoin.

10 Future Work

Applying the novel TCN architecture to other time series data such as weather patterns or traffic flow is an area of research that could show impressive results. This would likely involve some work to take in a new format of data, but could potentially rely on the same time series evaluations to make accurate predictions. Additionally, applying various other neural network architectures such as LSTMs could prove to be successful in predicting Bitcoin and gold prices. With an effective price prediction mechanism, an area of further research would be making our application being able to trade cryptocurrency in real-time. This would require research into effective trading policies. Finally, an effective cryptocurrency trading bot would be able perform arbitrage between markets.

11 Conclusion

In summary, we have utilized the TCN architecture to predict the prices of Bitcoin and gold. Random search was used to run our TCN model on the data with numerous different sets of uniformly sampled hyper-parameters. Our formulation was ineffective at accurately predicting Bitcoin prices and better at predicting gold prices but still not accurate enough to confidently invest in its predictions. We have analyzed the efficacy, as measured by average loss, of numerous hyper-parameters. We have shown that the intersection of these hyper-parameters plays the most important role in model performance and not any one hyper-parameter.

12 References

[1] Van Den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., ... Kavukcuoglu, K. (2016). WaveNet: A generative model for raw audio. SSW, 125.
[2] James Bergstra, Yoshua Bengio. (2012). Random Search for Hyper-Parameter Optimization. Journal of Machine Learning Research, 281-305.
[3] Shaojie Bai, J. Zico Kolter, Vladlen Koltun. (2018). An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling.
[4] Huisu Jang, Jaewook Lee. (2018). An Empirical Study on Modeling and Prediction of Bitcoin Prices With Bayesian Neural Networks Based on Blockchain Information. IEEE Access.
[5] Retrieved from https://datahub.io/core/gold-pricesresource-monthly on April 30th, 2019
[6] Retrieved from https://www.fxcm.com/uk/insights/what-causes-volatility-in-bitcoin/ on May 1st, 2019
[7] Retrieved from https://github.com/philipperemy/keras-tcn on May 1st, 2019
[8]CryptoCompare. (2019). Why Is the Price of Bitcoin so Volatile? Retrieved May 2, 2019 from https://www.cryptocompare.com/coins/guides/why-is-the-price-of-bitcoin-so-volatile/