Deep Learning and Ethereum Price Prediction

Deep Learning Systems Performance - Spring 2022 - Columbia University

Griffin Klett
Nathan Cuevas

Executive Summary

- We examined the feasibility and profitability of applying standard Deep Learning architectures to predicting the price of the Ethereum cryptocurrency, with Ethereum specific network parameters, macroeconomic influences, and public sentiment features, founded on the existing literature on Bitcoin price prediction.
- Our goal was to develop a profitable tradable strategy which could reliably return a profit over a long time-horizon and examine which DL architectures performed best during experimentation and testing.
- Our work adds value to the existing literature by demonstrating the translatability of findings on Bitcoin to generalized cryptocurrency modeling and underscores the value/benefit of domain specific feature selection

Problem Motivation

- In examining the existing literature on Cryptocurrency price prediction, we could not find any of the techniques applied to Ethereum, now the world's largest Blockchain by TVL (totally locked value). [So, a profitable strategy could make lots of money]!
- We felt that trying to apply similar techniques and architectures we'd read about on Bitcoin would be an interesting way to cement our understanding, and build something tangible
- DL is uniquely suited to be applied because of the ability to capture and learn hidden/abstract patterns from such a varied data factor set, and the inherent structure which has been shown to involve long memory that make traditional ML models less useful

Background Work

No.	Determinant	Category
21	Dow Jones industrial average	Macroeconomic environment
22	New York Stock Exchange Gold futures	
23	HuShen 300 index	
24	Korea composite index	
25	Singapore Straits Times index	
26	Standard & Poor's 500 index	

- A Comparative Study of Bitcoin Price Prediction Using Deep Learning. Ji et al. https://www.mdpi.com/2227-7390/7/10/898
- Deep Learning Methods for Modeling Bitcoin Price: Lamothe-Fernández et al. https://www.mdpi.com/2227-7390/8/8/1245
  - Built on top of Ji et al's research by using macroeconomic and social features
- Cryptocurrency forecasting with deep learning chaotic neural networks. Lahmiri et al. https://www.sciencedirect.com/science/article/pii/S0960077918310233
  - First paper to examine and find structure of time series of digital currencies revealed fractal dynamics, long memory, and self-similarity
  - Validated use case of models which perform well on such datasets – LSTM, anything with memory cabalities, etc.

Background Work - Results

- Even after hyperparameter tuning and experimentation of window size, most models were found to not be slightly profitable, or within the margin of error [Figure 1]; Similarly, regression loss analysis was found to be worse than classification
- Many of the same features were identified to be relevant (block size, difficulty, cost per txn)
- More complex models were outperformed by DNN, CNN

<u>Figure 1: Profitability Analysis</u>

Table 13. Results of a profitability analysis. $m = 20$ was used for regression models and $m = 50$ for classification models. The log values of the major features, the sequential partitioning, and the first value-based normalization were used. For every model, the initial budget was 10,000 USD and the final value of investment is shown in the table.

	DNN	LSTM	CNN	ResNet	CRNN	Ensemble	SVM	Base	Random
regression	6755.55	8806.72	6616.87	7608.35	8102.71	5772.99	<u>**9842.95**</u>	–	–
classification	<u>**10877.07**</u>	**10359.42**	**10422.19**	**10619.98**	**10315.18**	**10432.44**	9532.43	**9532.43**	**9918.70**

<u>Figure 2: Sequences of Time Series as input</u>

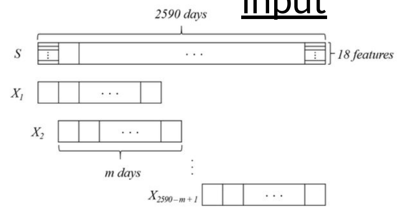


Figure 4. Data preprocessing. A total of $2590 - m + 1$ sequence data are generated from the whole dataset of 2590 days if m consecutive days are analyzed to predict the next Bitcoin price.

Technical Challenges

- Feature Selection: Unlike traditional assets, which all maintain similar financial reporting and have the same balance sheet input/outputs, each cryptocurrency may operate wildly differently than others (a la consensus mechanisms like proof of stake vs proof of work).
    - This makes domain knowledge of the relevant features which contribute to blockchain success important, and why applying similar techniques as on Bitcoin is not straightforward

- Hyperparameter tuning: Experimentation was necessary to understand the relationship that maximized the optimization between window (m) size and predictive performance

Dead zone b/c of trailing window

Technical Challenges

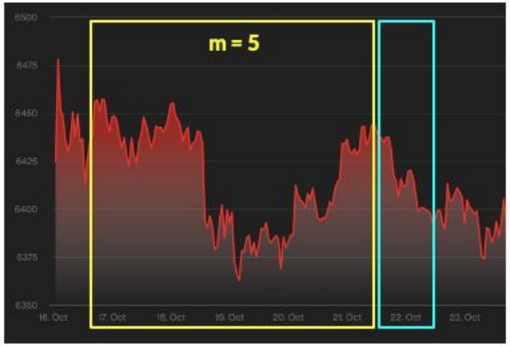
- Unbalanced Dataset:
  1000-fold increase in price and virtually all features (transaction rate, google-Index, etc) made life extremely difficult for our models, as we wanted to keep recent pricing data untouched for testing

Training set:

Test/Validation Set:

Approach

- Time series format of “last m-days” used as sliding window to make predictions and keep model from overfitting on seeing too much previous data. Each training point is an m-day x features array fed into model.
- Feature selection of Ethereum specific non-correlated features specific to network + macroeconomic + sentiments
- Keras models of DNN/CNN based off general structure described in Liu et al (num filters, kernel size) Interestingly, CNN did not perform better with more convolutions



Approach

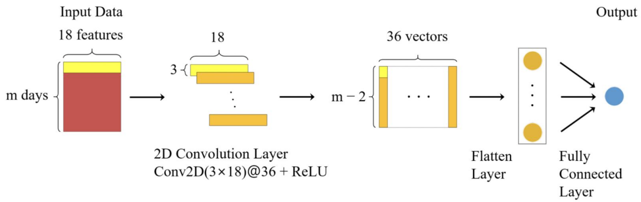
Feature set

Ethereum Price Prediction: Feature Engineering

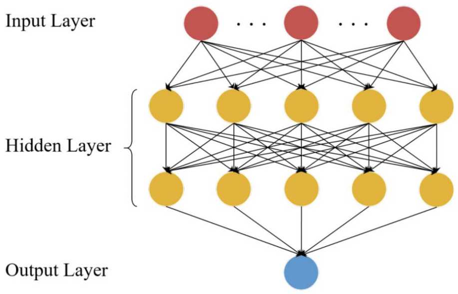
Feature	**Description/Category**
Difficulty	Mining problem difficulty (dynamically set by network based on success time of previous blocks)
Blocksize	Mb of transaction data included in each block
TxFeeUSD	Fee burned/paid to miners per block, paid by users transacting on the network
PriceBtcUsd	Bitcoin / USD trade ratio used for larger cryptocurrency market conditions
ERC20Transfer	User-User amount of ETH or ERCTokens transferred
ERC20Addr	Active Addresses
GTrendsEth	Search popularity ~ Proxy for public sentiment (Higher = better(
Gold	Market Sentiment / Inflation Hedge
S&P 500	Larger US Economy
Ticker/RIOT	Gaming / Bitcoin Mining company
Ticket/MSTR	Microstrategy BTC investment heavy fund

Solution Diagram/Architecture

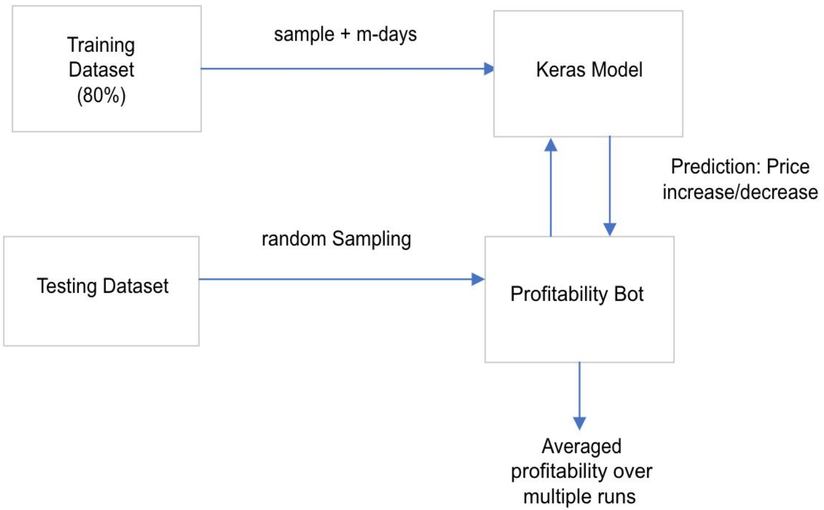
Convolutional Neural Networks (CNN)



Deep Neural Network (DNN)



Evaluation Architecture



Implementation Details

- Data consolidated from online sources such as etherscan.io (for Ethereum specific network params), G-Trends, Yahoo Finance via api and csv
- Models created in Keras, architecture for CNN/DNN structure after those described in Comparative Bitcoin paper
- Standard Train/Validation Split, normalized using day one values instead of min-max since price increased 5000% from 2015 (so standard minmax scaling barely affected the prices)

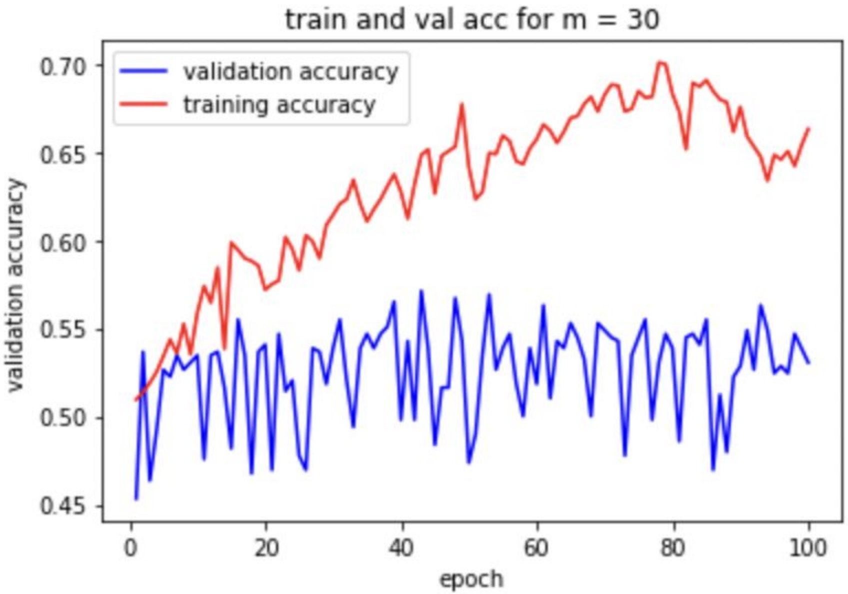
<u>CNN Model</u>

```
#Part 2
#CNN:
model = Sequential()
model.add(Input(shape=(m, num_features - 1, 1)))
model.add(Conv2D(filters=36, kernel_size=(3, 18), padding='same', activation='relu'))
model.add(Flatten())
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
```

Demo / Experiment Design

- Profitability bot given $10,000 and invests 5% of portfolio each day based on predicted price movement, or sells 5% if price is predicted to decrease. Assessed over subset of training dataset with both increases/decreases, validation holdout dataset.
- Multiple models with multiple runs of test datasets given to more accuracy simulate performance (averaged over 6 months performance * 10 random shuffles of sequence data)

Experimental Evaluation



- validation accuracy does not meaningfully improve over epoch, regardless of model architecture (CNN with m=30 shown for example)
- Dataset may be inadequate for generalization, or Ethereum exhibits otherwise non-predictable price behavior
- Increase in training accuracy due to learning dataset

Experimental Evaluation

Effects of Localization on Testing Accuracy

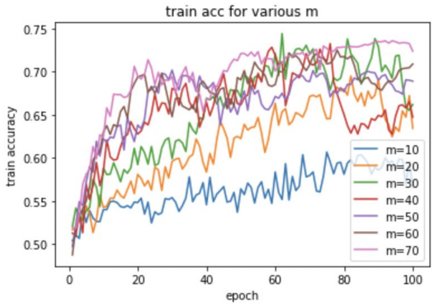
	m=10	m=20	m=30	m=40	m=50	m=60	m=70
DNN	53.6%	53.9%	53.7%	45.5%	45.7%	52.2%	56.9%
CNN	53.0%	55.9%	52.6%	53.9%	46.1%	53.2%	54.2%

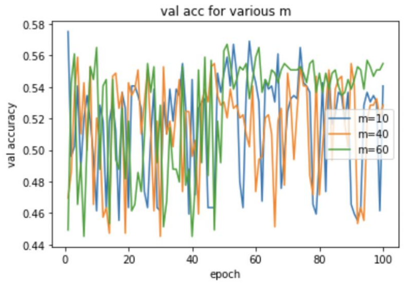
- Unlike Ji et al *m* didn’t seem to influence the accuracy much

Experimental Evaluation

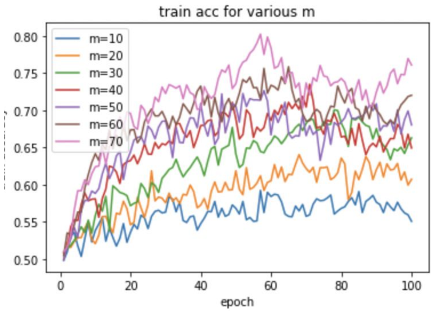
- Larger M values were learning the training dataset (larger m correlated with higher training accuracy), but unable to generalize on test.
- Similar behavior exhibited by DNN

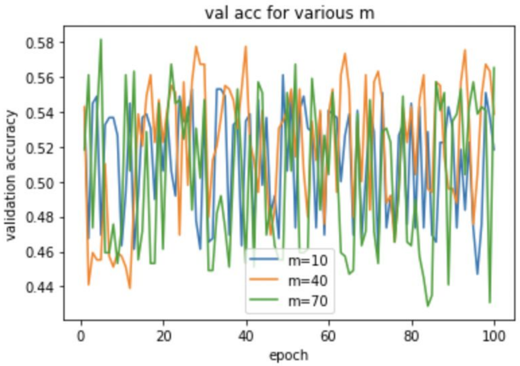
DNN performance across m-values





CNN performance across m-values





Experimental Evaluation

Trading bot ending investment given initial investment of $10,000

	DNN	CNN
m=10	$13,380.79	$32,730.82
m=40	$16,639.60	$9,848.46
m=70	$44,923.59	$65,950.31

Random
$6,995.83

- Each value is averaged over 5 runs
- For DNN m=70, buys almost every day; thest

Conclusion

- We were largely unsuccessful in building a profitable, deployable trading algorithm for ethereum prices; our models were unable to generalize to unseen data, regardless of how we normalized or sliced test/train, and regardless of how much localized information was fed into the model. Further, our model did not account for trading fees or associated liquidity costs with executing the algorithm.
    - E.g., unlike previous research on Bitcoin, our performance did not depend in any meaningful way on the size of m per training/testing
- The two possible consequences of these results suggest an improvement is needed in the data collection/exploration domain, or that Ethereum's market price is not subject to any discernible pattern by our DNN/CNN Models
- Further research could incorporate using some better scaling of the network or restricting the test/training time to the period after which ethererum had 1000x'd network size; this was impossible for us as the dataset remains too small but will be possible in the future.