Predicting Cryptocurrency Price Fluctuation Based on Twitter, Media, and Currency Data With an LSTM-RNN

Alex Fine

Motivation

Early in my freshmen year I noticed a fundamental problem with how many college students handle their money. Many students graduate thousands of dollars in debt, yet during college, what little money they do have often sits idle, and not invested. This issue often stems from the steep learning curve associated with investing well. My goal, was to give college students easy access to advanced investment opportunities, previously only available to the highly-rich and educated.

Summary

I built a dynamic RNN with multiple LSTM cells optimized to predict cryptocurrency price fluctuation on temporal price data with high granularity. The network is trained on aggregate Twitter sentiment, media sentiment, and past currency performance. The network is designed to easily predicts price change n minutes in the future. I found the highest accuracy and precision when predicting 5 minutes in the future.

Data

The RNN was trained on “sentiment-based cleansed tweets” (I removed all spam and promotional tweets by using a naive bias classifier), media sentiment and previous price data. In order to increase the training set size, I aggregated all data from all currencies to train a master RNN, and then created custom models for each individual currency.

Sentiment	Scale	Output
0.80	+ 3•5 + 4 + 0 = 19	15.2

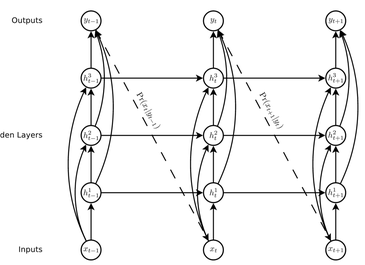
Features

Data: Volume, Price, net Twitter sentiment per minute
Response: All data logged per minute
Data Splitting: Training: 90%, Dev: 5%, Test: 5%
Data Size: Training: 2.38M, Validation: 217k

Models

This algorithm uses a dynamic RNN with 3 layers of LSTM cells, a state size of 5, and mini-batches. The core functionality was coded using TensorFlow’s nn libraries. The algorithm works by first creating an LSTM cell, which is passed into TensorFlow’s multi RNN cell, which is then passed into TensorFlow’s dynamic RNN.

$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$



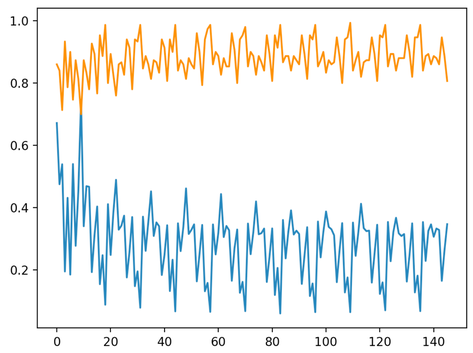
Results

The algorithm was most successful at predicting results 15 minutes in the future, when the most recent day of price fluctuation was high. The reported prediction accuracy was artificially high because the market skewed negative during the training period. I normalized such predictions, and reported both.

Model:	Normalized Train Accuracy	Normalized Test Accuracy	Actual Train Accuracy	Actual Test Accuracy
1 Min	49.86%	48.24%	82.97%	78.64%
5 Min	52.26%	50.51%	88.25%	84.86%
15 Min	55.44%	51.08%	88.23%	89.82%
1 Min T5	49.65%	48.54%	82.92%	79.06%
5 Min T5	53.44%	51.01%	88.95%	86.25%
15 T5	57.54%	51.69%	89.06%	90.05%

Discussion

This research conclusively mapped a loose relationship between aggregate market sentiment, as measured through twitter, and cryptocurrency price fluctuation. Predicting price change of low volume currencies was significantly more difficult than for high volume currencies. I attribute this to irregular data that can make training challenging. Additionally, I underestimated the highly influential roll that market behavior over training data plays on prediction data. Lastly, this research has made me question the implementation decision to concatenate the top 100 currencies. This concatenation led to irregular gradient descent, as depicted below.

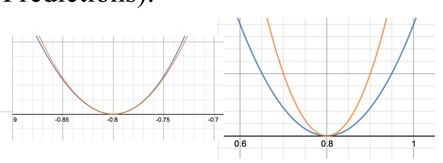


Future

Given more time, I would change the model, loss function, and implement more standard data normalization. To divert risk, I would like to implement the following custom loss function (Y = Lables, X = Predictions):

$$f(x) = \frac{1}{1 + e^{20x}}$$

$$L(x) = (Y - X)^2 \cdot e^{(f(P)Y - P)}$$



References:
[1] (image) Understanding LSTM units vs. cells https://stats.stackexchange.com/questions/241985/understanding-lstm-units-vs-cells
[2] (image) How does LSTM cell map to layers? https://stackoverflow.com/questions/45223467/how-does-lstm-cell-map-to-layers
[3] (image) https://twitter.com/MiklosDenkler/status/851922186101239808

Comprehensive references can be found in my paper. These references are just for images featured on the poster.