QBS
Vol. 37, No. 1, pp. 65-71 (2018)
https://doi.org/10.22283/qbs.2018.37.1.65
pISSN 2288-1344 eISSN 2508-7185 **Original Article**

Quantitative Bio-Science
Official Journal of the Institute of Natural Science at Keimyung University
Available Online at https://qbs.kmu.ac.kr:442/

Predicting Bitcoin Market Trend with Deep Learning Models

Yunbeom Seo, Changha Hwang*

Department of Data Science, Dankook University, Gyeonggido 16890, Korea
(Received May 8, 2018; Revised May 23, 2018; Accepted May 25, 2018)

ABSTRACT

As bitcoin attracts public attention as a new method of investment, the bitcoin market becomes a field of research that should be technically analyzed. Since bitcoin market is very uncertain, time series associated with bitcoin prices are complex, nonstationary and chaotic. In this paper, we attempt to predict bitcoin market trend using deep neural network (DNN) and four types of recurrent neural networks (RNNs). Our data set consists of fourteen input variables related to the bitcoin prices recorded daily from September 1, 2013 to August 31, 2017. Thirteen input variables are related to bitcoin prices, which are date, market price, high price, low price and closing price, and eight technical indicators derived from bitcoin prices. The other is keyword feature value obtained through text mining for social network services (SNS) data. Empirical study shows that DNN outperforms four RNNs in terms of specificity, precision and accuracy. Bidirectional RNN outperforms the other four deep learning models in terms of sensitivity. As a whole, DNN works quite well compared with RNNs.

Key words : Bitcoin, Deep learning, Deep neural network, Neural network, Recurrent neural network, Social network service, Text mining

1. Introduction

Efforts to reduce uncertainty have continued in many industrial fields for many years. See Kang and Hwang [1] for the uncertainty associated with recommender system. Lots of efforts also have been made in the stock market, since it has a lot of uncertainties due to the nationwide and worldwide factors. Under bounded rationality a large number of investors have utilized technical analysis to understand trend of the stock market and then obtain profits. This analysis is the most basic approach to stock market participants. Although it is not possible to correctly predict stock price, machine learning techniques have been actively applied to achieve better performance in predicting the fluctuations of stock prices.

* Correspondence should be addressed to Changha Hwang, Professor, Department of Data Science, Dankook University, Gyeonggido 16890, Republic of Korea. Tel: +82-31-8005-3261, Fax: +82-31-8021-7204, E-mail: chwang@dankook.ac.kr

Bitcoin has recently become a new investment instrument. Bitcoin, a block chain based digital currency, is currently leading the cryptocurrency market. Bitcoin market is very uncertain like stock market. Bitcoin prices change depending on asking price and trading volume like stock prices, and are freely determined in markets called bitcoin exchanges. Thus, time series associated with bitcoin prices are complex, nonstationary and chaotic.

Deep learning is the learning algorithm used for DNN composed of several layers. In this paper, we attempt to accurately predict fluctuations of bitcoin prices using deep learning models such as DNN, basic RNN, RNN with LSTM (long short term memory) cell, RNN with GRU (gated recurrent unit) cell and bidirectional RNN. The rest of this paper is organized as follows. Section 2 and section 3 briefly illustrates four deep learning models and technical indicators, respectively. Section 4 describes how to get quantitative data for analysis by applying text mining technique to traders opinion data obtained

through SNS. Section 5 and section 6 present experimental studies and conclusions, respectively.

2. Deep Learning Models

In this paper we use deep learning models for predicting fluctuations of bitcoin prices, since they have been actively applied to analyzing stock markets. In this section we illustrate four deep learning models.

2.1 DNN

DNNs are neural networks composed of one input layer and one output layer, and multiple hidden layers in between. DNNs are distinguished from shallow neural networks with single hidden layer by their depth. In DNNs, each layer of nodes trains on a distinct set of features based on the previous layer's output. In other words, each layer performs specific types of sorting and ordering in a process that some refer to as feature hierarchy. The further you advance into the neural net, the more complex the features your nodes can recognize, since they aggregate and recombine features from the previous layer. Thus, DNNs use sophisticated mathematical modeling to process data in complex ways. One of the key uses of these DNNs is dealing with unlabeled or unstructured data.

In classification problem, the weight and bias parameters of DNNs are estimated by obtaining the optimal solution to minimize the cross-entropy loss function between the estimated output and the target output. The parameters are updated using a backpropagation (BP) algorithm, until the cross-entropy loss function converges to the defined threshold value. In general, the disadvantages of the BP algorithm using a gradient descent method are known as follows. Firstly, it is sensitive to the initial values of weights and outliers of data set. Secondly, it has possibility for being stuck in local minima, Thirdly, it has tendency to overfitting for training data. Finally, it has slow computing speed.

Huge speed boost comes from using a faster optimizer than the regular gradient descent optimizer. In this paper we use Adam optimization [2] with a good initialization strategy for the weight parameters based on Glorot and Bengio [3]. For the experiment, we use DNN with 5 hidden layers, each of which has of 100 hidden nodes, and uses the ReLU function as the activation function. We also use the Adam optimizer as the optimizer. We set the mini-batch size to 10 and the number of epoch to 100. We may use deep learning algorithm proposed by Hinton et al. [4] for the DNN.

2.2 Standard RNN

RNNs are neural networks for processing sequential data. The standard RNN was developed by Elman [5]. This allows it to exhibit dynamic temporal behavior for a time sequence. Unlike standard neural networks, it has the RNN block applies something called as a recurrence formula to the input vector and also its previous state. We can process a sequence of vectors by applying a recurrence formula at every time step. The backpropagation through time (BPTT) algorithm is commonly used to learn RNNs. Contrary to standard neural networks, RNNs are characterized by the ability of encoding longer past information, thus very suitable for sequential models. The BPTT extends the traditional BP algorithm to suit the architectures of RNNs.

For the experiment, we use the standard RNN with 60 hidden nodes. We use the hyperbolic tangent function as the activation function. We set the learning rate to 0.01, the sequence length to 10 and the number of epoch to 250.

2.3 RNN with LSTM cell

Standard RNN has the vanishing gradient problem as disadvantage. To avoid this problem, the use of LSTM cell was introduced by Hochreiter and Schmidhuber [6]. For simplicity, we call it LSTM-RNN. It greatly reduces the multiplication of small gradients. The LSTM cell is a specifically designed unit that will help reduce the vanishing gradient problem sufficiently to make RNNs more useful for long-term memory tasks in practical settings. The way it does so is by creating an internal memory state which is simply added to the processed input, which greatly reduces the multiplicative effect of small gradients. The time dependence and effects of previous inputs are controlled by an interesting concept called a forget gate, which determines which states are remembered or forgotten. Two other gates, the input gate and output gate, are also featured in LSTM cells.

For the experiment, we use LSTM-RNN of dimension 60. We use the hyperbolic tangent function as the activation function. We set the learning rate to 0.01, the sequence length to 10 and the number of epoch to 200. We use 1 for the initial value of bias of forget gate.

2.4 RNN with GRU cell

To avoid the vanishing gradient problem the use of GRU was

introduced by Cho et al. [7]. For simplicity, we call it GRU-RNN. GRU is a gating mechanism in RNNs. The basic idea of using a GRU to learn long-term dependencies is the same as in a LSTM. A GRU has two gates, which are a reset gate and an update gate. The reset gate determines how to combine the new input with the previous memory, and the update gate defines how much of the previous memory to keep around. Similar to LSTM-RNN, GRU-RNN performs well on polyphonic music modeling and speech signal modeling. However, GRU-RNN exhibits better performance on smaller datasets than LSTM-RNN.

For the experiment, we use GRU-RNN of dimension 100. We use the sigmoid function as the activation function and the Adam optimizer as the optimizer. We set the learning rate to 0.01, the sequence length to 7 and the number of epoch to 200. We use 1 for the initial value of bias of forget gate.

2.5 Bidirectional RNN

Bidirectional RNN was introduced by Schuster and Paliwal [8] to increase the amount of input information available to the network. Bidirectional RNN is an extension of a RNN that focuses on the fact that the output value at a particular time step may also be affected by the future time step as well as the past time step. By the way, standard neural network and time delay neural network have limitations on the input data flexibility, since they require their input data to be fixed. Standard RNN also has restrictions as the future input information cannot be reached from the current state. On the contrary, bidirectional RNN does not require its input data to be fixed. Moreover, its future input information is reachable from the current state. The basic idea of bidirectional RNN is to connect two hidden layers of opposite directions to the same output. Thus, the output layer can get information from past and future states.

For the experiment, we use bidirectional RNN with basic structure. We use the sigmoid function as the activation function and the Adam optimizer as the optimizer. We set the mini-batch size to 7 and the number of epoch to 4.

3. Technical Indicators

In the stock market, technical analysis is an analysis methodology for forecasting the direction of stock prices through the study of past market data, primarily price and volume. In technical analysis, a technical indicator is a mathematical calculation based on historic stock price or volume that aims to forecast stock market direction. Typical technical indicators commonly used in the stock market are moving average, Bollinger band, relative strength index (RSI), moving average convergence divergence (MACD) and stochastic oscillator. In the paper we are going to use these technical indicators for technical analysis in the bitcoin market. See for details Murphy [9]. In this section we illustrate these indicators.

3.1 Moving average

A moving average is a calculation to analyze data points by creating series of averages of different subsets of the full data set. A moving average is commonly used with time series data to smooth out short-term fluctuations and highlight longer-term trends. It is often used in technical analysis of financial data. The averages are then joined to form the moving average line. The investment strategy using moving averages is simple, but it is the most commonly used technical analysis because of the advantage of being able to observe the overall price flow.

The moving average can be classified into simple moving average (SMA), weight moving average (WMA) and exponential moving average (EMA). In general, although the nearest price value affects the predicted value of the next period more, the SMA has the disadvantage that the same weight is given to the present and past price values. To overcome this disadvantage, we use the WMA, which is a method of assigning a low weight to old price values and a high weight to recent price values when calculating an average value. When the nth day has weight w_n in an n-day WMA, WMA is defined as

$$WMA(n) = w_n p_n + w_{n-1} p_{n-1} + \cdots + w_2 p_2 + w_1 p_1, \qquad (1)$$

where $\sum_{i=1}^{n} w_i = 1$ and $0 \le w_1 \le w_2 \le \cdots \le w_{n-1} \le w_n \le 1$. In the paper we use the 5-day WMA for closing bitcoin prices.

3.2 Bollinger band

The Bollinger bands indicator is a popular technical analysis tool developed by John Bollinger in the early 1980's. In general, Bollinger bands are plotted two standard deviations above and below a 20-day SMA. In other words, Bollinger bands consist of three bands which are plotted in relation to security prices. The middle band is usually a SMA set to a

period of 20 days. However, in this paper we use a 10-day SMA. The short-term trend seems well served by the 10-day calculations. The SMA then serves as a base for the upper and lower bands. The upper and lower bands are used as a way to measure volatility by observing the relationship between the bands and price. The upper and lower bands are set to two standard deviations away from the SMA. Three bands associated with Bollinger bands are defined as

$$\begin{aligned}&\text{Upper Band} = \bar{p} + 2\sigma\\ &\text{Middle Band} = \bar{p}\\ &\text{Lower Band} = \bar{p} - 2\sigma\end{aligned} \tag{2}$$

where $\bar{p} = \frac{1}{10}\sum_{i=1}^{10} p_i$ and $\sigma = \sqrt{\frac{1}{10}\sum_{i=1}^{10}(p_i - \bar{p})^2}$.

Two typical indicators derived from Bollinger bands are *%B* and bandwidth. In the paper, we consider these two indicators as input variables. The indicator *%B* tells us where we are within the bands. The bandwidth (BW) tells us how wide the bands are. These are defined as

$$\begin{aligned}&\%B = (C - \text{Lower Band}) / (\text{Upper Band} - \text{Lower Band})\\ &\text{BW} = (\text{Upper Band} - \text{Lower Band}) / \text{Middle Band},\end{aligned} \tag{3}$$

where *C* is the closing price of current day.

3.3 RSI

RSI helps the traders identify the price continuation and reversal. It is price momentum indicator and a kind of oscillator that provides a complete picture of the market by analyzing the velocity of the price movements. It compares the internal strength of the securities and not the relative strength of the different securities. In general, 9-day, 14-day or 25-day RSI is calculated. In the paper we use the 9-day RSI for closing bitcoin prices. It is calculated by using the following formula

$$RSI = 100 - 100/(1 + RS), \tag{4}$$

where relative strength (RS) is defined as $RS = Average\ Gain / Average\ Loss$.

When average gain is greater than the average loss, RS will be greater than one and RSI will increase always. If the average gain is less than the average loss, RS will be less than one and RSI will decline. Usually RSI is measured on 0 to 100 scales. If the RSI is above 70, it is an indication of the over bought position. When the RSI value is less than 30, it is indicative of the oversold position. During the over bought positions, the traders have to sell the security but in oversold position the traders have to buy the securities.

3.4 MACD

MACD is a trend-following momentum indicator that shows the relationship between two EMAs of prices. The MACD is calculated by subtracting the 26-day EMA from the 12-day EMA. A 9-day EMA of the MACD, called the MACD signal, is then plotted on top of the MACD, functioning as a trigger for buy and sell signals. MACD is all about the convergence and divergence of the two EMAs. Convergence occurs when the EMAs move towards each other. Divergence occurs when the EMAs move away from each other. The shorter EMA is faster and responsible for most MACD movements. The longer EMA is slower and less reactive to price changes in the underlying security. In the paper we use MACD and MACD signal as input variables.

The MACD line oscillates above and below the zero line, which is also known as the centerline. These crossovers indicate that the 12-day EMA has crossed the 26-day EMA. The direction depends on the direction of the EMA cross. Positive MACD indicates that the 12-day EMA is above the 26-day EMA. Positive values increase as the shorter EMA diverges further from the longer EMA. This means upside momentum is increasing. Negative MACD values indicate that the 12-day EMA is below the 26-day EMA. Negative values increase as the shorter EMA diverges further below the longer EMA. This means downside momentum is increasing.

3.5 Stochastic oscillator

The stochastic oscillator is a momentum indicator comparing the closing price of a security to the range of its prices over a certain period of time. The sensitivity of the oscillator to market movements is reducible by adjusting that time period or by taking a moving average of the result. This method attempts to predict price turning points by comparing the closing price of a security to its price range. The 5-day stochastic oscillator in a daily time frame is defined as follows:

$$\begin{aligned}&\%K = (C - L5) / (H5 - L5)\\ &\%D = \text{the 3-day SMA of } \%K\end{aligned} \tag{5}$$

where H5 and L5 are the highest and lowest prices in the last 5 days respectively, while %D is the 3-day SMA of %K (the last 3 values of %K). In the paper we use %K and %D as input variables.

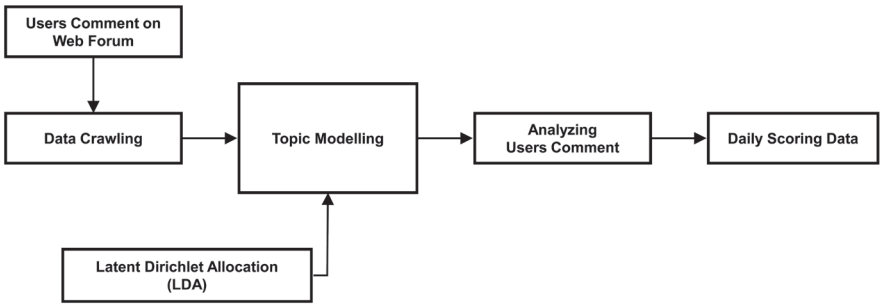

Fig. 1. Text mining process for SNS data.

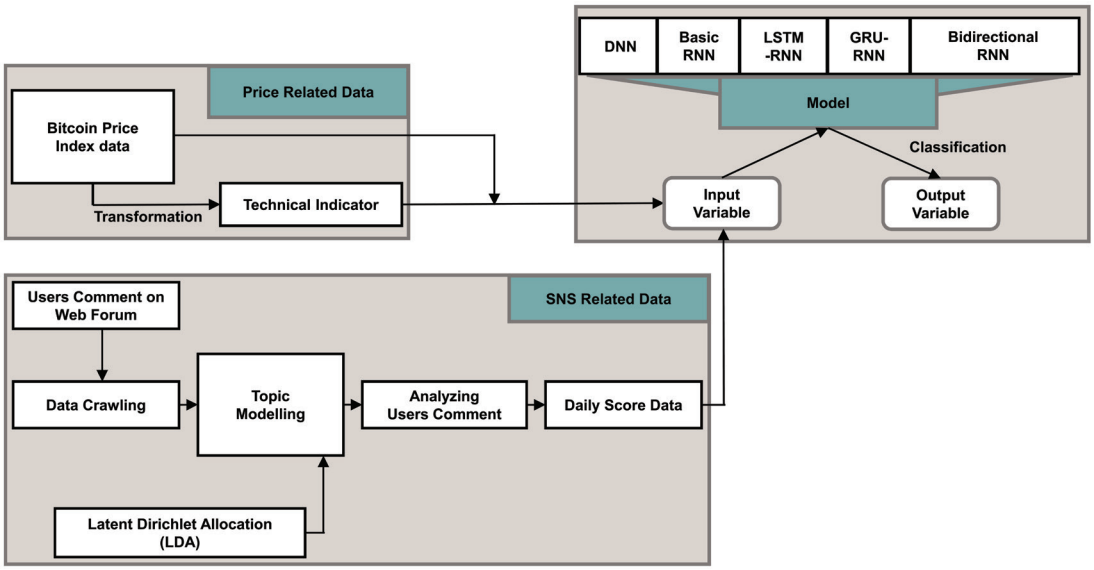

Fig. 2. Experimental process.

4. Keyword Feature Value for SNS Data

Due to the nature of the bitcoin that is traded online, a lot of information on the bitcoin is generated in real time on SNS. Traders inevitably make decisions based on a lot of information on SNS. In the paper, text mining analysis is conducted on opinion data of SNS through bitcoin forum which is a bitcoin related SNS. A similar analysis was conducted in Kim et al. [10]. Besides technical indicators related to bitcoin prices, qualitative data, such as opinions of traders, are converted into numerical values through text mining. Then the related variable is used as an input variable. The reason for using SNS data is that we want to use as another input variable the correlation between bitcoin price fluctuations and keywords on SNS.

The process of converting SNS data into numerical values is shown in Fig. 1. First, we perform data crawling task using Python for bitcoin comments from September 1, 2013 to August 31, 2017 on the bitcoin forum. Then, crawling data is subjected to a topic modeling process based on a latent Dirichlet allocation (LDA) algorithm. The LDA is applied to retrieve information from the texts. This method allows to discover hidden topics in documents and can be successfully used in sentimental analysis. We extract new scoring data related to fluctuations of bitcoin prices through topic modelling. The newly extracted scoring data is related to keyword feature value, which is actually used as an input variable.

Now we take a closer look at the sentimental analysis for traders opinions. If bitcoin prices are expected to rise, online opinion would be positive. Conversely, if bitcoin prices are expected to decline, it will be negative due to uneasiness. If the fluctuation of bitcoin price is uncertain, opinion may be uncertain. Thus, each of these opinions is classified into positive, negative or indifferent sentiment class through text min-

Table 1. Prediction of bitcoin price fluctuation based on bitcoin prices

Models	Measure			
	Sensitivity	Specificity	Precision	Accuracy
DNN	80.57851%	**72.72727%**	**78.31325%**	**77.04545%**
Basic RNN	97.78761%	33.00493%	61.90476%	67.13287%
LSTM-RNN	93.82716%	37.63441%	66.27907%	69.46387%
GRU-RNN	82.70677%	35.58282%	67.69231%	64.80186%
Bidirectional RNN	**99.64413%**	19.17211%	60.15038%	63.46719%

Table 2. Prediction of bitcoin price fluctuation based on bitcoin prices and SNS data

Models	Measure			
	Sensitivity	Specificity	Precision	Accuracy
DNN	83.88430%	**71.71717%**	**78.37838%**	**78.40909%**
Basic RNN	94.32314%	26.50000%	59.50413%	62.70396%
LSTM-RNN	86.77686%	37.43316%	64.22018%	65.26807%
GRU-RNN	91.41631%	29.59184%	60.68376%	63.17016%
Bidirectional RNN	**99.28826%**	21.56863%	60.78431%	64.34868%

ing. Class membership is expressed as a 3-dimensional vector. For example, class membership vector is $\theta=(0.2, 0.7, 0.1)^t$, then such opinion is negative.

Keyword feature value retrieved through text mining is defined as the value of an opinion on a specific day. Since there could be many opinions on the fluctuation of bitcoin price every day, keyword feature values are computed as follows. First, we compute the sum of daily opinions classified into each of positive, negative and indifferent sentiment classes. We assign $+1$, -1, 0 to an opinion classified into positive, negative and indifferent sentiment classes, respectively. If the vector of daily total numbers of opinions classified into each class is $(21, 10, 15)^t$ on a specific day, then the related keyword feature value is 21. If the vector is $(7, 15, 6)^t$ on a specific day, then keyword feature value is -15. If the vector is $(5, 6, 14)^t$ on a specific day, then keyword feature value is 0. In the paper this keyword feature value is used an input variable.

5. Experimental Results

The experimental process is shown in Fig. 2. Our data set consists of one output variable and fourteen input variables related to the bitcoin prices recorded for 1,461 working days from September 1, 2013 to August 31, 2017. Fourteen input variables are composed of five basic variables directly associated with bitcoin prices, eight technical indicators derived from bitcoin prices, and one keyword feature value obtained from SNS data. Five basic variables are date, market price, high price, low price and closing price. Technical indicators and keyword feature value are illustrated in Section 3 and Section 4. The direction of bitcoin price fluctuation is set to two directions of going up and going down. Output variable represents these two directions labeled as 1 and 0. The whole dataset is partitioned into training and test datasets at a ratio of 7 : 3.

Experimental results are shown in Table 1 and Table 2. Tables 1 and 2 illustrate the classification results for the direction of bitcoin price fluctuation based on thirteen input variables without keyword feature value and fourteen input variables with keyword feature value, respectively.

As seen from Tables 1 and 2, DNN shows the best performance in terms of specificity, precision and accuracy for both cases. Bidirectional RNN shows the best performance in terms of sensitivity. Prediction results are very similar for both cases. Three types of RNNs show poor performance except for sensitivity. These results illustrate DNN model predicts quite well the fluctuations of bitcoin prices. As a whole, prediction results when the bitcoin price based data are only used are better than those when the bitcoin price based data and the SNS data are used together.

6. Conclusion

In this paper, we have studied the prediction accuracy of

fluctuations of bitcoin prices by using several deep learning models. We have observed DNN shows the best performance in terms of specificity, precision and accuracy for both cases. Bidirectional RNN shows the best performance in terms of sensitivity. The deep learning models used in this paper show the prediction accuracy is not higher than that in previous studies related to the fluctuation predictions of stock prices. We think it is due to the fact that the existing researches on stock price prediction have had a lot of training data well reflecting KOSPI. It is also because transactions are made without interruption according to a lot of information generated online every second, and the number of investors investing for returns is growing rapidly.

In order to more accurately predict the fluctuations of bitcoin prices we need to develop new deep learning algorithms which overcomes the above limitations, and also need the refinement of DNN models.

Acknowledgements

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (NRF-2016R1D1A1B03931617).

References

1. Kang S, Hwang C. Deep recommender system using purchase history data. Quantitative Bio-Science 2017;36:45-50.
2. Kingma DP, Ba JL. Adam: A method for stochastic optimization. International Conference on Learning Representations. San Diego; 2015. https://arxiv.org/abs/1412.6980
3. Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks. International Conference on Artificial Intelligence and Statistics; 249-256. Sardinia, Italy; 2010.
4. Hinton GE, Osindero S, Teh YW. A fast learning algorithm for deep belief nets. Neural Comput 2006;18:1527-1554.
5. Elman J. Finding structure in time. Cognitive Science 1990; 14:179-211.
6. Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput 1997;9:1735-1780.
7. Cho K, Van Merrienboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation. Conference on Empirical Methods in Natural Language Processing; 1724-1734. Doha, Qatar; Oct. 2014.
8. Schuster M, Paliwal KK. Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing 1997;45:2673-2681.
9. Murphy J. Technical analysis of the financial markets. New York: New York Institute of Finance; 1999.
10. Kim YB, Kim JG, Kim W, Im JH, Kim TH, Kang SJ. Predicting fluctuations in cryptocurrency transactions based on user comments and replies. PLoS ONE 2016;11:e0161197.