



**Classification of Cryptocurrency Price Trend   
Using Gradient Boosting and LSTM**

Do-Hyung Kwon\*, Ju-Bong Kim\*\*, Ju-Sung Heo\*, Chan-Myung Kim\*\*\* and Youn-Hee Han\*\*



**Abstract**

In this paper, we apply machine learning and deep learning to classify Cryptocurrency price data. We collected price data through Bithumb's API, and preprocess and normalize it. Our research shows that classifying the trend of Cryptocurrency using these models is superior to traditional methodologies, even for highly volatile time series data. In particular, a grid search based cross validation technique is applied to find the most suitable parameter for the prediction model. In general, Long Short-Term Memory (LSTM), which is used in this paper, can solve complex and long time-lag tasks that are never solved by previous iterative network algorithms. At the end of this article, our studies show that the LSTM model outperforms Gradient Boosting, a general algorithm for machine learning.

**Keywords**

Time series analysis, Classification, Gradient Boosting, LSTM



※ This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (<http://creativecommons.org/licenses/>by-nc/3.0/) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

Manuscript received Month 06, 20XX; accepted Month 06, 20XX.

**Corresponding Author:** Youn-Hee Han (yhhan@koreatech.ac.kr)

\* Interdisciplinary Program in Creative Engineering, Korea University of Technology and Education, Cheonan, Republic of Korea ({}@koreatech.ac.kr)

\*\* Department of Computer Engineering, Korea University of Technology and Education, Cheonan, Republic of Korea ({}@koreatech.ac.kr)

\*\*\* Advanced Technology Research Center, Korea University of Technology and Education, Cheonan, Republic of Korea (cmdr@koreatech.ac.kr)

**1. Introduction**

From the past, stock price prediction is a difficult problem to solve. Various attempts and studies have been made to predict stock price scientifically [1]. However, accurate price prediction is impossible yet. In the stock market, thus, it is common to generate profit by using an algorithm trading instead of a direct investment method. Grove et al. investigated 136 cases and concluded that mathematical models yield better results than humans, or 94%, which is better than humans [13].

Numerous Cryptocurrencies have been developed recently, starting with Bitcoin, which implements a distributed ledger, and a Cryptocurrency market has been formed. Especially, Bitcoin is attractive in a variety of fields such as economics, cryptography, and computer science due to the unique nature of the combination of encryption technology and currency units [3]. The Cryptocurrency market has a relatively short history compared to the stock market. Fundamentally, both stock price data and Cryptocurrency price data have arbitrary characteristics as time series data, but the latter has higher volatility, and the prices of other Cryptocurrencies are affected by the price of certain Cryptocurrency.

The Cryptocurrency market is different from the existing stock market and has many new features relatively [3]. Existing techniques require hand-crafted features and expert knowledge of the field that were expensive to create. So, in addition to existing stock market analysis techniques, it is necessary to apply new analytical techniques suitable for the Cryptocurrency market.

Although many related studies have been conducted in the field of stock price, there is not yet much research on Crypto-currency price prediction that has received much attention recently. There are few studies related to Cryptocurrency price data. Reported paper is so far [7], [8]. However, refer-ence [7] and [8] are far from individual Cryptocurrency trend predictions in that they propose a portfolio management approach that determines which Cryptocurrency should be invested heavily.

There has been an increasing trend to apply deep learning techniques recently to stock price prediction. Reference [9] constructed a model that estimates the stock price of Sam-sung Electronics using 15 input features and showed a per-formance of about 50%. Reference [10] shows about 56% performance using deep learning-based model. In particular, Reference [10] uses simple stock price data such as open prices, close prices, high prices and low prices. Stanford University's study [11] used a variety of machine learning models from September 1, 2008 to November 8, 2013 to predict stock prices and showed performances of 44.52% to 58.2%. Reference [12] attempted to select an optimal trading policy through a neural-based stock trading system.

The purpose of this paper is to use machine learning and deep learning to predict the trend of stock prices without manual work. In detail, goal of this article is to construct a model to classify the direction (up or down) for a Cryptocurrency price after the th unit time based on a certain unit time by learning the past price changes of Cyptocurrencies. In this paper, we study classifying the stock price data using Artificial Intelligence and then, determining whether the stock price will increase or not rather than predict it. In particular, we try to explore the Cryptocurrency price data applying existing analysis methods, and apply machine learning and deep learning to classify the trend of Cryptocur-rencies.

The rest of this paper is structured as follows: Section II briefly reviews time series, blockchain, Cryptocurrency, and conventional approach such as ARIMA, and Gradient Boost-ing, XGBoost, and LSTM, which are machine learning and deep learning models. Section III discusses collecting Cryptocurrency price data and preprocessing, data structure. Section IV describe the model we treated and techniques for finding optimal parameters. Section V outlines experimental results. Section VI concludes the article.

2. Literature Review

## 2.1 Time Series

Time series consist of measurements collected over time in various fields such as text analysis data, ECG data, stock price data, temperature data, merchandise sales volume, and exchange rate data. It is a data whose values change with the lapse of time. When collected together, it constitutes what is known as a time series [4].

Malhotra et al. [6] defined that time series is represented as follows:

X ***=*** (1)

Each vector in X is an array whose dimension is . The vector is described as follows.

(2)

Each point is equivalent to the input variables.

We can ask about time series data: First, does the measured value show a certain pattern or tendency over time? Second, Is there repeated patterns in the time series data for one-year cycle or season? Third, is there any outliers in time series data? For example, if the time series data shows a specific pattern by season, it can be said to have seasonality. If time series data has these characteristics, we can apply a proven model such as Seasonal ARIMA.

4.1 Gradient boosting

Gradient boosting (GB) is an ensemble learning methods and classify by combining the outputs from individual trees [XX]. GB builds trees one at a time, where each new tree helps to correct errors made by previously trained tree. GB exploits the boosting scheme, which focuses step by step on difficult data samples by strengthening the impact of the successful classification.

3. Data Treatments

This section describes the process of collecting cryptocurrency price data and reconstructing them for use as the training data. The data is collected by Bithumb API. Initially, they are mixed with various abnormal data, so that they need to be preprocessed.

3.1 Data Collection and Preprocessing

We collected cryptocurrency price values every 10 minutes. The collected data includes the five features: open price, close price, high price, low price, and volume at each time epoch (that is, every 10 minutes). In our work, we consider seven kinds of cryptocurrency to be used in this work; BTC (Bitcoin), ETH (Ethereum), XRP (Ripple), BCH (Bitcoin Cash), LTC (Litecoin), DASH (Dash) and ETC (Ethereum Classic), and the one fiat currency, KRW (Korean Won). The price and volume data is collected from 0:00 on June 9, 2017 to 09:00 on May 8, 2018 through Bithumb API.

**Table 1.** Example of the collected raw BTC price data.

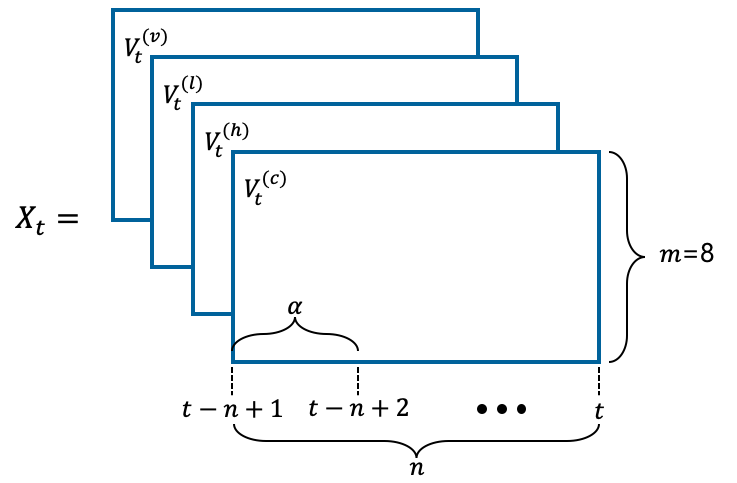
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Index | Time epoch | Open price | Close price | High price | Low price | Volume |
| 1 | 2017-06-02 08:50:00 | 3121000 | 3130000 | 3142000 | 3121000 | 99.295 |
| 2 | 2017-06-02 09:00:00 | 3130000 | 3142000 | 3143000 | 3130000 | 105.998 |
| 3 | 2017-06-02 09:09:00 | 3142000 | 3141000 | 3147000 | 3132000 | 101.861 |
| 4 | 2017-06-02 09:10:00 | 3142000 | 3143000 | 3146000 | 3141000 | 88.417 |
| 5 | 2017-06-02 09:20:00 | - | - | - | - | - |
| 6 | Missing | Missing | Missing | Missing | Missing | Missing |
| 7 | Missing | Missing | Missing | Missing | Missing | Missing |
| 8 | 2017-06-02 09:50:00 | 3100000 | 3200000 | 3222000 | 3190000 | 182.123 |
| 9 | 2017-06-02 09:59:00 | 3200000 | 3100000 | 3112000 | 3088000 | 98.762 |
| 10 | 2017-06-02 10:10:00 | 3100000 | 3099000 | 3110000 | 3087000 | 177.615 |

Table 1 shows an example of the collected raw BTC price data. The collected data includes 1) normal data (Indices 1, 2, 4, 8, and 10 in Table 1), 2) missing data, 3) empty data, and 4) non-matching data. Missing data means data that has not been collected in a time interval of 10 minutes (Indices 6 and 7 in Table 1). Empty data is collected in a time interval of 10 minutes but the data is empty (Index 5 in Table 1). Non-matching data is data that does not fit perfectly every 10 minutes (Indices 3 and 9 in Table 1)[[1]](#footnote-1).If certain data is empty or missed, then the values of the previous normal data are copied to fill the value of the abnormal data. For example, in the case of missing data, the previous data are copied and the time information is properly inserted, and the case of empty data is also processed in the similar manner. In case of non-matching data, if there are already normal data collected at 10 minutes intervals, the non-matching data, of which time is the closest to the normal data, is removed (Index 3 in Table 1). Otherwise, the time information of the data is updated with the time adjusted to correct 10 minutes intervals (Index 9 in Table 1). Table 2 shows the pre-processed data. It has one less record than the raw data of Table 1, since the record of which index is 3 in Table 1 is removed.

**Table 2**. The preprocessed BTC price data for the raw data of Table 1.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Index | Time epoch | Open price | Close price | High price | Low price | Volume |
| 1 | 2017-06-02 08:50:00 | 3121000 | 3130000 | 3142000 | 3121000 | 99.295 |
| 2 | 2017-06-02 09:00:00 | 3130000 | 3142000 | 3143000 | 3130000 | 105.998 |
| 3 | 2017-06-02 09:10:00 | 3142000 | 3143000 | 3146000 | 3141000 | 88.417 |
| 4 | 2017-06-02 09:20:00 | 3142000 | 3143000 | 3146000 | 3141000 | 88.417 |
| 5 | 2017-06-02 09:30:00 | 3142000 | 3143000 | 3146000 | 3141000 | 88.417 |
| 6 | 2017-06-02 09:40:00 | 3142000 | 3143000 | 3146000 | 3141000 | 88.417 |
| 7 | 2017-06-02 09:50:00 | 3100000 | 3200000 | 3222000 | 3190000 | 182.123 |
| 8 | 2017-06-02 10:00:00 | 3200000 | 3100000 | 3112000 | 3088000 | 98.762 |
| 9 | 2017-06-02 10:10:00 | 3100000 | 3099000 | 3110000 | 3087000 | 177.615 |

3.2 Training and Testing Data



**Fig. 3.** Data structure of price tensor at the time epoch (: the window size, : the unit of time, : the number of cryptocurrencies)

Our training and testing data are generated to the price tensor using the preprocessed data. The price tensor is defined as similar to the one introduced in [8]. Price tensor at the epoch consists of four 2-dimentional matrixes , , , and that represent close price, high price, low price, and transaction volume for every unit of time , respectively. Open price is not included in a price tensor because it usually equals to close price of the previous time epoch. In our study, the unit of time will be 10, 30, or 60 minutes.

The close price for the epoch of the -th cryptocurrency is denoted by . On the other hand, is the high price data, is the low price data, and is the transaction volume of the epoch . The -th cryptocurrency represents KRW (Korean Won) and its tensor value is always 1, regardless of epoch , since we assume that it plays the role of standard currency.

(14)

Price tensor at a time epoch includes the past price and volume data for the unit of time . We call the window size. If and , price tensor includes the price data during minutes (that is, the price data for every 150 time epoch). Let the number of cryptocurrencies including the base currency, and it is set to (KRW, BTC, ETH, XRP, BCH, LTC, DASH and ETC) in this study. Figure 3 shows the data structure of the price tensor at the time epoch for the window size , the unit of time , and the eight cryptocurrencies (=8). When the operator is defined as element-wise vector division, the four matrixes , , , and for the epoch and are as follows:

(15)

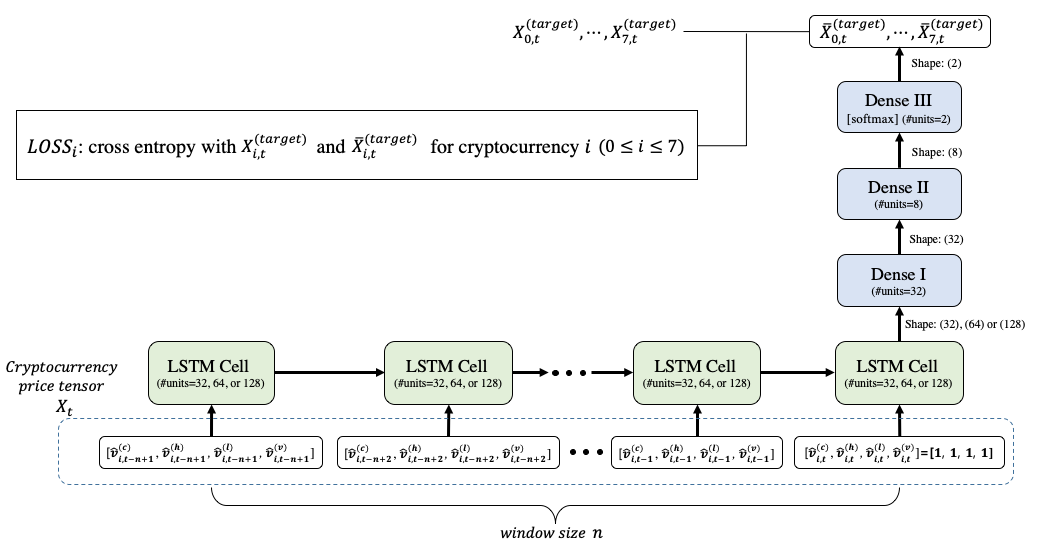
Equation (15) represents that all the values are divided by the corresponding last price or last volume, so that they are well normalized.

On the other hand, for a price tensor at the epoch , the target (label) data is expressed for a cryptocurrency () as follows.

For a price tensor, there are eight target data. It means that we will predict the price trend individually for each of the eight cryptocurrencies from the past price data of the total eight ones. In equation (17), is the close price of unit of times elapsed from to for the cryptocurrency . If exceeds than , (price-up) is given, and 0 (price-down) otherwise. That is, is the rate of price increase after unit of times. In our study, and are fixed to 6 and 1, respectively. Therefore, if the unit of time is set to 30 minutes, the target value of a cryptocurrency at a time epoch is 1 if the cryptocurrency’s price goes over 1% after 180(= minutes, and 0 otherwise.

4. Training and Validation

4.1 Model Structure



**Fig. 4.** Proposed LSTM model structure(the windows size is and , , , for)

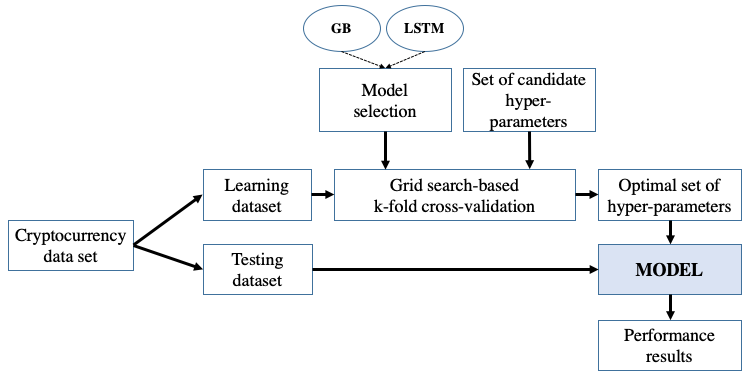
In our work, we use the long short-term memory (LSTM) model for the binary classification of cryptocurrency price trend. The LSTM model [22] is a recurrent neural network model. The LSTM model determines whether the weight value is maintained by adding cell states in an LSTM cell. The LSTM model can accept arbitrary length of inputs, and it can be implemented flexibly and in various ways as required. The state obtained from an LSTM cell is used as an input to the next LSTM cell, so that the state of an LSTM cell affects the operation of the subsequent cells. The final target output at the end of the sequence represents a label classifying the price trend (up or down). The LSTM model has the ability to remove or add information to the cell state, carefully regulated by structures called gates. Gates are a way to optionally let information through. The LSTM model is more persistent than existing RNN because it is possible to control long-term memory.

As shown in Figure 3, our LSTM model consists of one LSTM layer, which has LSTM cells, and three dense layers. For a time epoch , the price tensor is used for the input for the LSTM layer. The number of hidden units of an LSTM cell will be 32, 64, or 128. The proper number of the units will be chosen by the model tuning process and explained in the next subsection. On the other hand, the numbers of hidden units at the three dense layers are fixed to 32, 8, and 2, respectively. Except for the last dense layer (Dense III) using ‘softmax’, all LSTM and other dense layers (Dense I and II) use the activation function selected by the model tuning process, which be explained in the next sub-section.. Since the last dense layer’s activation is ‘softmax’, a last output vector has just two values, which add up to 1, and they can be interpreted as probabilities for price up and down trends. For , the last outputs are compared with the target data for the price tensor . As the loss function , we use the cross entropy which indicates the distance between and .

4.2 Model Tuning

A machine learning model has various hyper-parameters that determine the network structure (e.g., number of hidden units) and how the models are trained (e.g., type of optimizer). The performance of a model can vary considerably according to the selected set of hyper-parameters. In this study, we use the grid search method [30] to find the optimal hyper-parameters for our cryptocurrency price dataset by trying every possible combination of hyper-parameters based on the dataset. We also verify the validity of the model by performing k-fold cross-validation [31] in addition to finding the optimal hyper-parameters.

To demonstrate the superiority of the proposed LSTM model, we also used the gradient boosting (GB) classifier model, which is known as a traditional machine learning model showing good performance in various fields. For fairness, we also perform the same model tuning process for the GB classifier model.



**Fig. 4.** Overall model tuning process with grid search-based k-fold cross-validation

Figure 4 shows the overall process of finding the optimal set of hyper-parameters and validating the GB and LSTM models with the optimal ones. First, we separate the cryptocurrency price dataset into learning and test data. Next, -fold cross-validation based on grid search is performed. In -fold cross-validation, the training dataset is randomly partitioned into equal sized sub-datasets. Of the sub-datasets, a single sub-dataset is retained as the validation data for testing the model, and the remaining sub-datasets are used as training data. The cross-validation process is then repeated times, with each of the sub-datasets used exactly once as the validation data. The results can then be averaged to produce a single set of optimal hyper parameters. Then, the selected model is again trained by using the optimal hyper-parameters. Finally, the model performance is measured using the test data. The performance evaluation results will be given in Section 5.

For the hyper-parameters of LSTM model, we consider 1) number of hidden units of an LSTM cell, 2) parameter initializer, 3) activation type, 4) dropout rate, and 6) optimization type. The number of hidden units of an LSTM cell is the dimensionality of the last output space of the LSTM layer. The parameter initializer represents the strategy to initialize the LSTM and Dense layers’ weight values. The activation type represents the type of activation function which produces non-linear and limited output signals inside the LSTM and Dense I and II layers. Further, the dropout rate indicates the fraction of the hidden units to drop for the transformation of the recurrent state in the LSTM layer. Finally, the optimization type designates the optimization algorithm to tune the internal model parameters so as to minimize the cross entropy loss function. In the LSTM model, the candidate values used to perform the grid search for the hyper-parameters are listed in Table 1.The table also lists an example of the optimal hyper-parameter values found by our model tuning process.

**Table 3.** The candidate and the optimal set of hyper-parameters for the LSTM model

|  |  |  |
| --- | --- | --- |
| **Hyper-parameter name** | **Hyper-parameter values** | **Example of the optimal hyper-parameter values** ( and ) |
| number of hidden units of an LSTM cell | {32, 64, 128} | 64 |
| parameter initializer | {normal, he\_normal, glorot\_normal} | Glorot\_normal |
| activation type | {relu, tanh, sigmoid} | Relu |
| dropout rate | {0.0, 0.2, 0.3, 0.4} | 0.2 |
| optimization type | {SGD, RMSProp, Adagrad, Adam} | RMSProp |

On the other hand, Table II shows the candidate values used to perform the grid search for the GB hyper-parameters, and also lists the optimal hyper-parameter values. For the GB model, we consider 1) maximum depth, 2) maximum features, 3) number of estimators, 4) subsamples, and 5) minimum samples for split. For the details of these hyper-parameters, refer to [XX, XX].

**Table 4.** The candidate and the optimal set of hyper-parameters for the GB model

|  |  |  |
| --- | --- | --- |
| **Hyper-parameter name** | **Hyper-parameter values** | **Optimal hyper-parameter values** ( and ) |
| maximum depth | {4, 5, 6} | 5 |
| maximum features | {all, sqrt, log2} | all |
| number of estimators | {50, 75, 100} | 75 |
| subsamples | {0.8, 0.9, 1.0} | 0.9 |
| minimum samples for split | {2, 3, 4} | 2 |

5. Performance Evaluation Results

In this section, we evaluate the performance of the LSTM model’s binary classification and compare the LSTM model’s prediction performance against the GB model’s one.

5.1 Evaluation Environment

Our evaluation task is executed on Ubuntu 16.04 LTS with 32 GB of RAM and two GPU cards (NVIDIA GTX 1080Ti 11 GB). We used Tensorflow-gpu 1.8 and Keras 2.2.0 operated with Python 3.6 According to the process shown in Figure 4, the 3-fold cross-validated grid search is first performed for the cryptocurrency price tensor data set (that is, k=3) and the optimal hyper-parameters are found through each model tuning. The model training is performed using the found optimal hyper-parameters listed in Tables 5 and 6. For such model training, the number of learning epochs is set to 200.

5.2 Performance Metrics

**Table 5.** An example of binary confusion matrix for the LSTM model evaluation ( and )

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Predictive | |
|  |  | Positive | Negative |
| Actual | Positive | 599  (True Positive: TP) | 3 (False Negative: FN) |
| Negative | 1 (False Positive: FP) | 4191 (True Negative: TN) |

An unambiguous and thorough way to present the prediction results of a deep learning model is to use a confusion matrix. Table 5 is an example of the binary confusion matrix by the LSTM model evaluation when the window size is 10. In the binary confusion matrix, the true positive (TP) indicates the cases in which the actual label is positive (that is, price-up) and the model prediction is also positive correctly. The false negative (FN) indicates the cases in which the actual label is positive, but the model prediction is negative (that is, price-down) incorrectly. The false positive (FP) indicates the cases in which the actual label is negative (that is, not RDP), but the model prediction is positive incorrectly. Finally, the true negative (TN) indicates the cases in which the actual label is negative and the model prediction is also negative correctly.

As shown in Table 5, there is an imbalance on the number of actual target data (the number of actual positive is XXX, while the number of actual negative is XXX). The accuracy measurement can be misleading when there is such imbalance. A model can predict the target of the majority for all predictions and achieve a high classification accuracy, and the model becomes not useful. To overcome the problem of accuracy measurement, we compute the additional measurements to evaluate the LSTM and GB models: recall, precision, and f1-score [32]. They are defined by using the confusion matrix as follows:

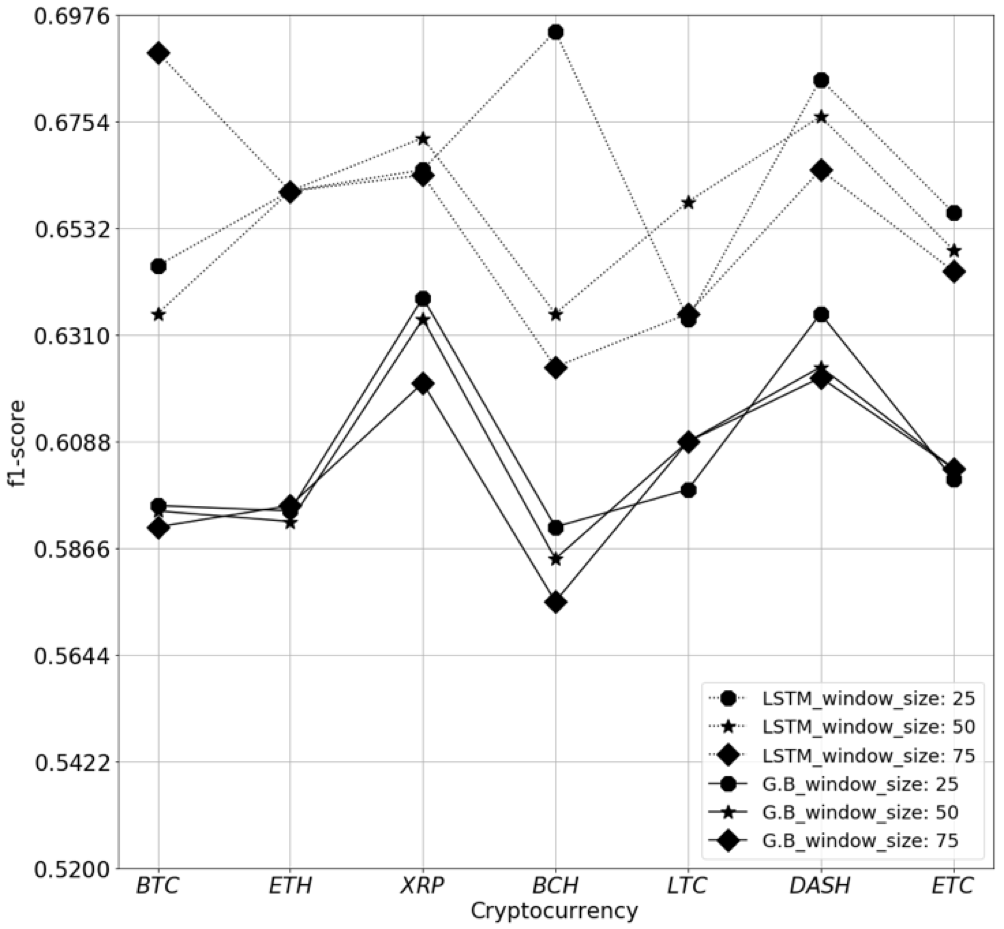
(1)

(2)

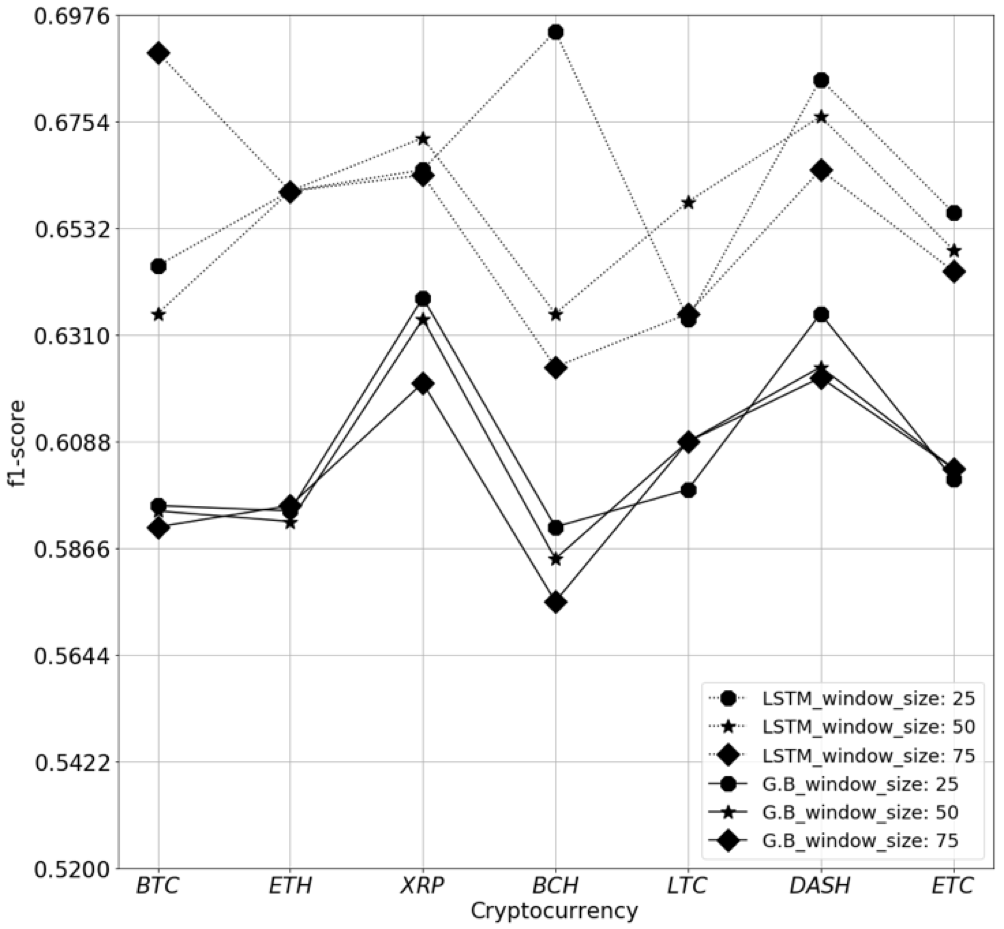
(3)

The f1-score represents the harmonic mean of precision and recall, and indicates the classification performance of a model relatively accurately. It is expressed in the range of 0 to 1, where the best value is 1. We compute the recalls, precisions, and f1-scores, then finally get a single f1-score measurement for performance evaluation of the LSTM and GB models.

5.3 Evaluation Results



**Fig. 5.** Performance comparison of the LSTM and GB models in terms of eight cryptocurrencies and window size (unit of time is fixed to 10)



**Fig. 6.** Performance comparison of the LSTM and GB models in terms of eight cryptocurrencies and unit of times (window size is fixed to 25)

6. Conclusions

Stock price prediction has been extensively studied in the economic field. However, the study of Cryptocurrency price prediction has only recently been studied. In this paper, classification experiments were conducted to determine whether the price of Cryptocurrency will increase or decrease using machine learning and deep learning models. In particular, we performed an experiment to find the optimal prediction model using the grid search method. Despite the high volatility of the Cryptocurrency price data, using the deep learning method resulte in a higher F1-score than the machine learning method.

The results of the Gradient Boosting model show that XRP is relatively predictable while BCH is hard to predict. Also, it can be seen that there is no significant difference in the performance of the model in time unit. That is, further consideration of past data has a negative impact on model predictions. And we have shown through experiments that LSTM has a better F1 score than Gradient Boosting. Despite using a simple LSTM model, the average F1 score is approximately 0.643. On the other hand, the F1 score value of the gradient boosting is about 0.597. Therefore, the deep learning method is more suitable than the machine learning method when classifying the price data of Cryptocurrency which is time series data with high volatility.

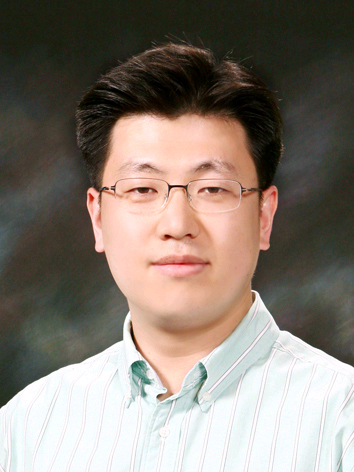
In the future research, we will conduct the same experiment using time series data from various domains such as ECG. In particular, we will continue to explore the data structure of the time series used to learn the model.

Acknowledgement

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2018R1A6A1A03025526).

**References**

1. V. Kolici, A. Herrero, and F. Xhafa, "On the performance of Oracle grid engine queuing system for computing intensive applications," *Journal of Information Processing Systems*, vol. 10, no, 4, pp. 491-502, Dec. 2014. E. Kita, Y. Zuo, M. Harada and T. Mizuno, “Application of Bayesian Network to stock price prediction,” Artificial Intelligence Research, vol. 1, no. 2, Dec. 2012.
2. Satoshi Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System”
3. H.S. Jang and J.W. Lee, "An Empirical Study on Modeling and Prediction of Bitcoin Prices With Bayesian Neural Networks Based on Blockchain Information," IEEE Access, vol. 6, pp. 5427–5437, 2017.
4. John Gamboa, “Deep Learning for Time-Series Analysis,” arXiv:1701.01887 [cs.LG], Jan. 2017.
5. D.-H. Kwon, J.-S. Heo, J.-B. Kim, H.-K. Lim and Y.-H Han, “Correlation Analysis and Regression Test on Cryptocurrency Price Data,” KIPS, Republic of Korea, vol. 25, no. 1, pp. 346-349, May. 2018.
6. Malhotra, P., Vig, L., Shroff, G., Agarwal, P., “Long short term memory networks for anomaly detection in time series,” European Symposium on Artificial Neural Networks, vol. 23, pp. 89-94, April. 2015.
7. Z. Jiang and J. Liang, “Cryptocurrency portfolio management with deep reinforcement learning,” arXiv:1612.01277 [cs.LG], May. 2017.
8. Z. Jiang, D. Xu, and J. Liang, “A Deep Reinforcement Learning Framework for the Financial Portfolio Management Problem,” Journal of Machine Learning Research, arXiv:1706.10059 [q-fin.CP], Jul. 2017.
9. Y. Song, J. W. Lee, and J. Lee, “Performance Evaluation of Price-based Input Features in Stock Price Prediction using Tensorflow,” KIISE Transactions on Computing Practices, vol. 23, no.11, pp.625-631, 2018.
10. Y. Song and J. Lee, “A Design and Implementation of Deep Learning Model for Stock Predictions using TensorFlow,” Processing of Korea Information Science Society Conference, pp.799-801, June. 2017.
11. Y. Dai and Y. Zhang, “Machine Learning in Stock Price Trend Forecasting,” Stanford University, 2013.
12. J. W. Lee and J. M. O, “Artificial Intelligence: Integrated Multiple Simulation for Optimizing Performance of Stock Trading Systems based on Neural Networks,” KIPS Journal, vol. 14B, no. 2, pp.127-134, Feb. 2007.
13. W.M. Grove, D.H. Zald, B.S. Lebow, B.E. Snitz and C. Nelson, “Clinical versus mechanical prediction: a meta-analysis,” Psychological Assessment, vol. 12, no. 1, pp. 19-30, Mar. 2000.
14. P. Hall, B.U. Park, and R.J. Samworth, “Choice of Neighbor Order in Nearest-neighbor Classification,” Annals of Statistics, vol. 36, no. 5, pp. 2135-2152, Oct. 2008.
15. C.J.C. Burges, “A Tutorial on Support Vector Machines for Pattern Recognition,” Data Mining and Knowledge Discovery, vol. 2, no. 2, pp. 121-167, Jun. 1998.
16. Y.D. Kim, K.H. Kim, and S.H. Song, “Comparison of Boosting and SVM,” Journal of the Korean Data and Information Science Society, vol. 16, no. 4, pp. 999-1012, Dec. 2005.
17. J.H. Jung and D.K. Min, “The study of foreign exchange trading revenue model using decision tree and gradient boosting,” Journal of the Korean Data and Information Science Society, vol. 24, no. 1, pp. 161-170, 2013.
18. A. Natekin and A. Knoll “Gradient Boosting Machines, a Tutorial,” Front Neurorobot, vol. 7, no. 21, Dec. 2013.
19. S. Kar, S. Saha, L. Khaidem, and S. R. Dey, “Predicting the Direction of Stock Market Price Using Tree Based Classifiers,” The North American Journal of Economics and Finance, Jul. 2018.
20. B. Gorman, “A Kaggle Master Explains Gradient Boosting,” [Online], Available:http://blog.kaggle.com/2017/01/23/a-kaggle-master-explains-gradient-boosting, Jan. 2017.
21. K. P. Murphy, Machine Learning: A Probabilistic Perspective. Cambridge, MA, USA: MIT Press, 2012.
22. R.A. Carmona, "Time Series Models: AR, MA, ARMA, & All That," vol. , Springer, New York, NY, 2004, pp. 239-309.
23. M. Falk, F. Marohn, R. Michel, M. Macke, D. Hofmann, C. Spachmann and S. Englert, "A First Course on Time Series Analysis Examples with SAS," University of Wuerzburg, March, 2011.
24. G. Kitagawa, "Introduction to Time Series Modeling," CRC Press, April. 2010.
25. J.-S. Heo, D.-H. Kwon, J.-B. Kim, Y.-H. Han and C.-H. An, "Prediction of Cryptocurrency Price Trend Using Gradient Boosting," KIPS Transactions on Software and Data Engineering, vol. 7, no. 10, pp. 387-396, Oct. 2018
26. S. Hochreiter and J. Schmidhuber, "LONG SHORT-TERM MEMORY," Neural Computation, vol. 9, no. 8, pp. 1735-1780, Nov. 1997
27. C. Olah, “Understanding LSTM Networks,” [Online], Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/, Aug. 2015
28. F. Chollet. Keras. [Online], Available: https://keras.io, 2015.
29. J. Bergstra and Y. Bengio. “Random search for hyperparameter optimization,” Journal of Machine Learning Research, vol. 13, pp. 281-305, Feb. 2012.
30. T. Fushiki, “Estimation of prediction error by using K-fold cross-validation,” Statistics and Computing, vol. 21, no. 2, pp. 137-146, 2011.
31. scikit-learn developers, [Online], Available: https://scikit-learn.org/stable/modules/generated/sklearn.model\_selection.RandomizedSearchCV.html

**Youn-Hee Han** / [http://orcid.org/0000-0002-5835-7972](http://orcid.org/0000-0002-4510-0385) (ORCID ID)

He received B.S. degree in Mathematics from Korea University, Seoul, Korea, in 1996. He received his M.S. and Ph.D. degrees in Computer Science and Engineering from Korea University in 1998 and 2002, respectively. From March 4, 2002 to February 28, 2006, he was a senior researcher in the Next Generation Network Group of Samsung Advanced Institute of Technology. Since March 2, 2006, he has been a Professor in the School of Computer Science and Engineering at Korea University of Technology and Education, CheonAn, Korea. His primary research interests include theory and application of mobile computing, including protocol design and mathematical analysis. Since 2002, his activities have focused on mobility management, media independent handover, and cross-layer optimization for efficient mobility support. His research topics also include mobile sensor/actuator networks, social network analysis, and deep learning. He has published approximately 150 research papers on the theory and application of mobile computing, and has filed 30 patents on ICT (Information and Communication Technology) domain. He has been serving as an editor for journal of information processing systems (JIPS) since August 2011. In addition, he has also made several contributions in IETF and IEEE standardization, and served as the co-chair of s working group in Korea TTA IPv6 Project Group.

1. The time information of the non-matching data is underlined. [↑](#footnote-ref-1)