Attention! Transformer with Sentiment on Cryptocurrencies Price Prediction

Huali Zhao, Martin Crane and Marija Bezbradica
School of Computing, Dublin City University, Dublin, Ireland
huali.zhao3@mail.dcu.ie, martin.crane@dcu.ie, marija.bezbradica@dcu.ie

Keywords: Transformer, Attention, Sentiment analysis, Cryptocurrency prediction

Abstract: Cryptocurrencies have won a lot of attention as an investment tool in recent years. Specific research has been done on cryptocurrencies' price prediction while the prices surge up. Classic models and recurrent neural networks are applied for the time series forecast. However, there remains limited research on how the Transformer works on forecasting cryptocurrencies price data. This paper investigated the forecasting capability of the Transformer model on Bitcoin (BTC) price data and Ethereum (ETH) price data which are time series with high fluctuation. Long short term memory model (LSTM) is employed for performance comparison. The result shows that LSTM performs better than Transformer both on BTC and ETH price prediction. Furthermore, in this paper, we also investigated if sentiment analysis can help improve the model's performance in forecasting future prices. Twitter data and Valence Aware Dictionary and sEntiment Reasoner (VADER) is used for getting sentiment scores. The result shows that the sentiment analysis improves the Transformer model's performance on BTC price but not ETH price. For the LSTM model, the sentiment analysis does not help with prediction results. Finally, this paper also shows that transfer learning can help on improving the Transformer's prediction ability on ETH price data.

1 INTRODUCTION

As a digital currency backed by cryptographic technology, cryptocurrency has held its place in investment portfolios (?). In 2020, the total assets under management of crypto hedge funds globally increased to nearly US$3.8 billion from US$2 billion the previous year. Around a fifth of hedge funds are investing in digital assets (21%). More than 85% of those hedge funds intend to deploy more capital into the asset class by the end of 2021 (?). Along with increasing interest in cryptocurrencies, the amount of research on cryptocurrency price prediction is growing (?) (?) (?) (?) (?) (?). Studies on applying statistical models and Recursive Neural Networks (RNN) models on Bitcoin(BTC) price have been done(?) (?) (?) (?). In the existing studies, the classic ARIMA model and various RNN models are the favorite when predicting future BTC price.

In addition to applying models on time series data, how public sentiment is driving cryptocurrency price is another popular topic. Studies found a strong correlation between public sentiment and and BTC price trend (?) (?) (?) (?) (?) (?).

However, among the existing cryptocurrency price prediction research, there is not much related to the Transformer (?) model. Both Long short term memory model (LSTM) and Transformer is powerful and efficient on Natural Language Processing(NLP). LSTM is also popular in time series forecasting field where Transformer is still new. In this paper, a study on how well the Transformer model can perform on BTC and ETH price data is performed. A comparison between the predictive performance of LSTM model and Transformer is done with 6 years of BTC price data and 5 years of ETH price data. We also investigated whether sentiment analysis can help on improving Transformer and LSTM prediction with sentiment data collected from Twitter.

This paper is organized as follows. In section 2, we review the related work done on time series forecasting using various models along with sentiment analysis. How data is collected and prepared for this study is illustrated in Section 3. Section 4 gives an overview of what Transformer is and how Transformer is utilized in this study. Results and findings are presented in Section 5 with conclusions in Section 6.

2 RELATED WORK

Time series modeling is widely utilized across domains like geography (?) and economics (?). Applying various Deep neural networks (DNN) for forecasting and pattern recognition on BTC price has been popular in recent years. Ji et al. (?) did a forecasting capability comparison on BTC price within DNN, LSTM (?), Convolutional neural networks(CNN), and deep residual networks (ResNet). Although there is no an overall winner in the competition, LSTM slightly outperforms in forecasting future BTC prices, while CNN outperforms others on indicating price moving direction. Facebook has created a regression model called PROPHET which is optimized for the business forecast tasks (?). Yenidogan et al. (?) proved success of this model in forecasting BTC future price by comparing PROPHET with ARIMA model. The result shows PROPHET outperforms ARIMA by 26% on R^2.

Sentiment has been shown to be a factor that impacts BTC future price. Guerra et al. (?) proved the correlation between BTC price and web sentiment (Twitter sentiment, Wikipedia search queries and Google search queries) by utilizing Support Vector Machine (SVM) model. By combining Fuzzy Transform on forecasting BTC price with Google trend data, the authors' study showed that web searches data can help on short-term BTC price prediction. Serafini et al. (?) composed a dataset which contains daily BTC weighted price, BTC volume, sentiment from Twitter and tweets volume, applied Auto-Regressive Integrated Moving Average with eXogenous input (ARIMAX) and LSTM-based RNN model on the data. They found that the linear model ARIMAX performs better than the LSTM-based RNN model on BTC price prediction. They also discovered out that the tweets sentiment instead of tweets volume is the most significant factor in predicting BTC price. Raju and Tarif's research (?) has also utilized sentiment analysis. They collected sentiment data from two sources: Twitter and Reddit. By applying both the LSTM and ARIMA model on a dataset composed of BTC price data and sentiment data, the authors found LSTM did better on the BTC price forecasting regression task. The study also indicated that combined sentiment data from different sources can improve the predicting result. Prajapati's (?) research compared CNN, (Gated Recursive Unit) GRU and LSTM model's performance on a dataset that composed by BTC's open, high, low, close, volume, Litcoin's close, volume, ETH's close, volume and sentiment data from Google news and Reddit. The result shows LSTM can give the lowest Root mean squared error (RMSE) on predicting BTC price.

Instead of looking at the regression problem, Kilimci et al. (?) is focused on BTC price moving direction classification problem. A comparison between deep learning architectures(CNN, LSTM and RNN) and word embedding models(Word2Vec, GloVe and FastText) on predicting BTC price moving direction using Twitter sentiment data is done. The research result shows that the word embedding model FastText (?) (?) achieves the best result with 89.13% accuracy. The performance order was FastText $>$ LSTM $>$ CNN $>$ RNN $\sim=$ GloVe $>$ Word2Vect.

Like FastText, Transformer (?) is popular for NLP tasks. Transformer is based on the multi-head attention mechanism (?) which allows the model to understand coherent relationships between the past tokens and the current token in NLP tasks. Based on the assumption that a time series is a sentence, a time point is a position in the sentence, and the data at the time point can be considered as the word in the position of the sentence. Under this assumption, the Transformer with multi-head attention can be utilized as a time series forecasting tool.

Li et al. (?) proved the assumption. The authors implemented a model with a dual attention layer to predict next time point public sentiment against P2P companies: Yucheng Group, Kuailu Group and Zhongjin Group. The time series that the model was applied on contains data points which are composed of micro blog post content which contains 0-140 Chinese words, author, pubtime, number of fans, and user category. LSTM, SVM, CNN and a model composed by two layers of LSTM with SVM were compared with the proposed model. The result shows that the proposed model with a dual attention layer was the winner. The study also suggested Transformer can capture long-term dependencies not captured by LSTM (?). They also proposed convolutional self-attention and sparse attention to further improve Transformer's performance by incorporating local context and reducing memory cost.

Inspired by the above studies, and considering lack of studies on Transformers prediction ability on cryptocurrency future price, this research explores Transformer's forecasting ability on BTC and ETH time series data with sentiment analysis.

3 DATA

3.1 Data Collection

6 years of BTC price data from 2015/01/01 to 2021/04/27 and 5 years of ETH data from 2016/05/09

to 2021/04/27 is collected from CoinAPI [1] for this study. The downloaded raw data contains the following information: time period start, time period end, time open, time close, price open, price high, price low, price close, volume traded, and trades count. For a single day, there are 8 data points collected with 3 hours intervals. In total, there are 18,454 BTC data points and 14,200 ETH data points.

To gather daily sentiment against BTC and ETH from Twitter, tweets are scraped using snscrape api [2]. 10,000 tweets are scraped for each day in the date range (2015/01/01 to 2021/04/27 for BTC, 2016/05/09 to 2021/04/27 for ETH). A scraped tweet contains the tweet text content, tweet id, user name, tweet language, date time when the tweet is posted, how many likes the tweet got and how many times the tweet is retweeted.

3.2 Data Preparation

3.2.1 Sentiment Analysis

In Pano and Kashef's study (**?**), 13 text preprocessing strategies and 4 sentiment analysis methods are compared. By applying the various techniques on scraped tweets using Tweepy [3], Pano and Kashef indicate that for text prepossessing, splitting sentences or removing Twitter-specific tags can improve the correlation of sentiment scores with Bitcoin prices. To get the sentiment score, the authors employed Valence Aware Dictionary and sEntiment Reasoner(VADER), a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media [4].

Following the above approach, in this study, the sentiment score is obtained for a tweet scraped from Twitter by 2 steps: 1) Cleaning the tweet text by removing Twitter handles, URLs, and special characters. 2) Using VADER to assign sentiment polarity score to the cleaned tweet. The polarity score is either 1 (positive), 0 (neutral) or -1 (negative). To get the sentiment score for a specific day, 10,000 tweets are scraped for the day, and the top 100 most liked and most retweeted tweets are selected for sentiment analysis using VADER. The final sentiment score for the day is the average score of the additive score of the select 100 tweets.

3.2.2 Final Dataset for The Experiment

From the raw BTC and ETH price data, the following columns are extracted respectively: time period start, open, high, low, close (OHLC) and volume traded. To normalize the data, MinMaxScaler from sklearn library is applied. The calculated sentiment score for each day is joined into the dataset to create the final dataset for this research. The overall data processing procedure illustrated in Figure 1.

The processed dataset is split into train and test datasets with a ratio of 80:20. The sliding window method is applied so the model can understand the coherence of the data points in the time series. The window size is set to be 240. The correspond output is the 241st data point.

4 METHODOLOGY

4.1 Overview

The design of the Transformer is shown in Figure 2. The architecture starts with Time2Vec layer that adds time embedding features to the input matrix. It followed by a stacked Transformer encoder layers that has a multi-head attention layer in the each of the encoder layer. Every multi-head attention layer is composed by 12 single-head attention layers. The final output is produced after the output of the stacked Transformer encoder layers passes through a Global average pooling layer, a Dropout layer and a Dense layer.

4.2 Model Design

4.2.1 Time Embedding - Position Encoding for Time Series

Positional encoding is required for Transformer to understand the absolute or relative position information of input words when the model is applied on NLP tasks. With time series inputs, time is arguably as essential as word position is for NLP. Periodical and non-periodical information in the dataset need to be understood by the model. Inspired by positional encoding, Kazemi et al. (**?**) created a vector representation of time called Time2Vec. This method is described as "a model-agnostic vector representation for time" in Kazemi's paper. It can be potentially used as an extra layer in any architecture (**?**). The idea behind Time2Vec is that time representation should contain periodic patterns and non-periodic patterns. It

[1] https://www.coinapi.io/

[2] https://github.com/JustAnotherArchivist/snscrape

[3] http://www.tweepy.org/

[4] https://github.com/cjhutto/vaderSentiment

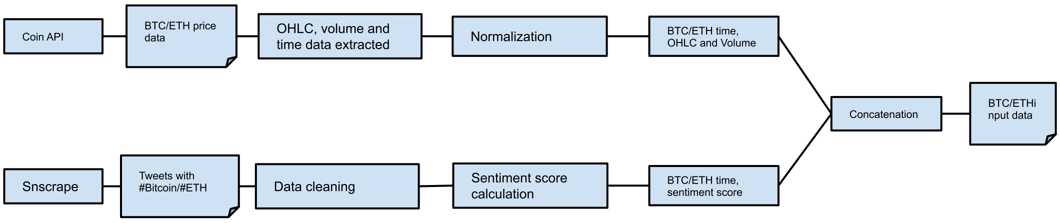
Coin API
BTC/ETH price data
OHLC, volume and time data extracted
Normalization
BTC/ETH time, OHLC and Volume
Concatenation
BTC/ETHi nput data
Snscrape
Tweets with #Bitcoin/#ETH
Data cleaning
Sentiment score calculation
BTC/ETH time, sentiment score

Figure 1: How data is scraped and processed.

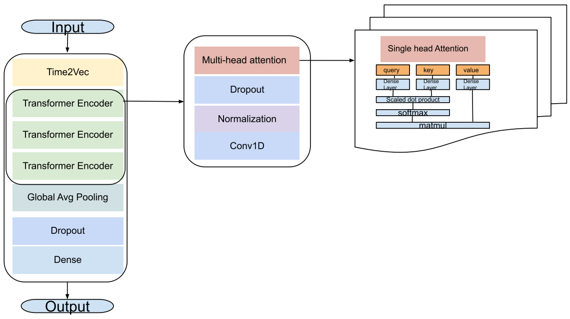

Figure 2: The overall system design of the Transformer for cryptocurrency price prediction

also should be invariant to time rescaling. In mathematical representation, Time2Vec can be represented by Equation 1 (**?**). Here, $\omega_i \tau + \varphi_i$ is a linear function which represents non-periodical components of the time series. ω is the slope of the linear function and φ is the interception. $f(\omega_i \tau + \varphi_i)$ is the periodical part where f is a sine function which helps to capture periodic behaviors without the need for feature engineering (**?**).

$$t2v(\tau)[i] = \begin{cases} \omega_i \tau + \varphi_i, & \text{if i=0} \\ f(\omega_i \tau + \varphi_i) & \text{if } 1 \leq \text{i} \leq \text{k} \end{cases} \tag{1}$$

After input data pass this Time2Vec layer, two outputs: periodical and non-periodically features are calculated. These features are concatenated with the input data to produce a new input matrix to feed Transformer model.

4.2.2 Single-head Attention - Transformer's Self Scaled-dot Attention

An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors (**?**). Query is calculated from the current token. Keys are calculated from past tokens. The query and the keys are compared to get attention weights for the values. The sum of the weights is 1.

The attention mechanism firstly creates three vectors: Q(query), K(key), and V(value) for each word embedding. These vectors are created by multiplying the input embedding x with three matrics W^Q, W^K and W^V, Equation 2, 3, 4. It then calculates the score for each word in the input sentence, divides the scores by the square root of the K dimension, softmax the dividend score, and finally multiples V vector with the softmax score, Equation 5. Using a scaled dot function helps in getting more stable gradients and softmax function makes sure the scores at each input sequence position will add up to 1.

$$Q = x * W^Q \tag{2}$$

$$K = x * W^K \tag{3}$$

$$V = x * W^V \tag{4}$$

$$Attention(Q,K,V) = Softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{5}$$

4.2.3 Multi-head Attention

Multi-head attention is a process of concatenating h single-head attention weights. Q, K and V are linearly projected h times. For each projected version of K, V and Q pairs, an attention layer is applied. Each attention layer produces output values in parallel for each version of projected K, V and Q. These outputs are concatenated and linearly projected to get the final value. By doing this multi-head attention, the model increases its ability to focus on different parts of the input sequence by taking in information from different representations jointly.

$$MultiHead(Q,K,V) = Concat(head_1, ..., head_h)W^o \tag{6}$$

where

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \tag{7}$$

5 RESULTS

5.1 Metrics

Mean Square Error (MSE), MAE (Mean Absolute Error) and Mean Absolute Percentage Error (MAPE) are selected as metrics for the model accuracy.

MSE is the average value of the squared difference between prediction and actual value. It gives variance of the residuals, Equation 8. The value can be drastically changed from a large outlier.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y})^2 \tag{8}$$

MAE is the average value of the absolute difference between prediction and actual value, Equation 9. It is average magnitude of error produced by models. It is not sensitive to outliers.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}| \tag{9}$$

MAPE shows the distance between estimation and reality in percentage, Equation 10. Compared with MAE, it is normalized by the actual value. It can be problematic when the actual value is 0.

$$MAPE = \frac{100}{n}\sum_{i=1}^{n}|\frac{y_i - \hat{y}}{y_i}| \tag{10}$$

5.2 LSTM vs Transfomer on BTC and ETH Prices Without Sentiment Data

As Table1 and Table2 show, applying the proposed Transformer model on the processed BTC and ETH price data does not achieve the same result as LSTM. It shows LSTM is better than the proposed model on BTC and ETH prices prediction. Like Ji (**?**), Mohan (**?**), and Raju (**?**)'s research, this study also proves LSTM is an outstanding model on regression problems.

5.3 Will Sentiment Scores Improve Transformer and LSTM's Performance?

After adding sentiment scores in BTC data, the proposed Transformer's model is improved by 0.001 on MSE, 0.1228 on MAPE and 0.01465 on MAE. Adding sentiment scores in ETH data does not get the same improvement. For the LSTM model, adding sentiment scores introduced more outliers on ETH future price prediction but no improvements on predictions with both BTC and ETH data, Table 5 and Table 6.

Table 1: Transformer vs LSTM on BTC OHLC price

Model	MSE	MAPE	MAE
Transformer	0.00137	0.18096	0.02900
LSTM	0.00033	0.04343	0.01310

Table 2: Transformer vs LSTM on ETH OHLC price

Model	MSE	MAPE	MAE
Transformer	0.15987	0.66011	0.33890
LSTM	0.00126	0.06651	0.02931

Table 3: Transformer's performance before and after adding sentiment scores in BTC data

Data	MSE	MAPE	MAE
BTC	0.00137	0.18096	0.02900
BTC+sentiment	0.00037	0.05816	0.01435

Table 4: Transformer's performance before and after adding sentiment scores in ETH data

Data	MSE	MAPE	MAE
ETH	0.15987	0.66011	0.33890
ETH+sentiment	0.16289	0.65803	0.34170

Table 5: LSTM performance before and after adding sentiment scores in BTC data

Data	MSE	MAPE	MAE
BTC	0.00033	0.04343	0.01310
BTC+sentiment	0.00032	0.04613	0.01346

Table 6: LSTM performance before and after adding sentiment scores in ETH data

Data	MSE	MAPE	MAE
ETH	0.00126	0.06651	0.02931
ETH+sentiment	0.00586	0.03316	0.01656

5.4 Transfer learning

An interesting finding in this study is that by applying the Transformer model trained with BTC data on predicting ETH future price, better results are achieved than by applying the Transformer model trained with ETH data only. Table 7 and Table 8 show the big jump on the metrics. It can be seen in Figure 3 that ETH price movement has a very similar trend as BTC price movement. The calculated Spearman's correlation between the BTC close price and ETH close price is 0.732. It means the BTC close price is highly correlated with ETH close price. As leading cryptocurrencies in the market, ETH and BTC datasets have similar nature. Predicting the future price of ETH and

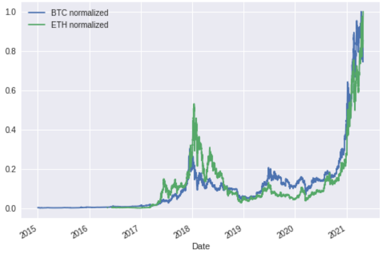

Figure 3: Normalized BTC and ETH price

BTC can be considered a related problem. Therefore the knowledge Transformer model got when learning from BTC data can be applied to predicting ETH price. Due to the limited amount of ETH data, the ETH dataset has 4245 fewer data points when compared with the BTC dataset. The Transformer model learns more context from the BTC dataset.

Table 7: Transfomer model on ETH OHLC price

Model	MSE	MAPE	MAE
Trained with ETH data	0.15987	0.66011	0.33890
Trained with BTC data	0.00081	0.06093	0.02157

Table 8: Transfomer model on ETH time series with sentiment scores

Model	MSE	MAPE	MAE
Trained with ETH data	0.16289	0.65803	0.34170
Trained with BTC data	0.00567	0.06697	0.02412

6 CONCLUSION

In this study, we have implemented a Transformer model with stacked Transformer encoder layers. In contrast with the standard Transformer model applied in NLP, the proposed model has a Time2Vec layer to implement time embedding. In each of the Transformer encoder layers, there is one multi-head attention layer which is concatenated by 12 single-head attention. We applied the proposed model to the problem of BTC and ETH price prediction. We also compared the forecasting ability between the proposed Transformer model and LSTM model. The result shows that the LSTM model outperforms the proposed model on predicting BTC and ETH future price with or without sentiment scores data. With sentiment scores added in the BTC dataset, there is an obvious improvement on the proposed model's prediction result. For ETH future prediction, there is no improvement. Applying the LSTM model on both dataset with sentiment scores, there is no performance improvement but more outliers produced with ETH dataset. An interesting finding on the proposed model is that the model trained with BTC data can give better prediction results on ETH future price than the model trained with ETH data.

Future work can start from the following aspects. First, the sentiment data collection. The collected sentiment data from this study is the latest 10,000 tweets of each day. When calculating the sentiment score, the popularity of the tweets is considered. Compared with news from finance websites or newspapers with more influence, the sentiment could be different from what is collected from Twitter. In future study, sentiment from CoinDesk and Bloomberg might provide a different angle of how people view the price movement. Second, sentiment analysis. Instead of using sentiment scores as a feature, the news text or tweets text can be converted to a feature vector. The sentiment analysis can be carried out as a model layer which takes in the text. Third, improving model performance by introducing temporal pattern attention (**?**) or sparse attention (**?**).