# Bitcoin price prediction using ARIMA and LSTM

Yiqing Hua[1]

[1]Jacobs School of Engineering, University of California San Diego, 92037, US

**Abstract**. The goal of this paper is to compare the accuracy of bitcoin price in USD prediction based on two different model, Long Short term Memory (LSTM) network and ARIMA model. Real-time price data is collected by Pycurl from Bitfine. LSTM model is implemented by Keras and TensorFlow. ARIMA model used in this paper is mainly to present a classical comparison of time series forecasting, as expected, it could make efficient prediction limited in short-time interval, and the outcome depends on the time period. The LSTM could reach a better performance, with extra, indispensable time for model training, especially via CPU.

## 1 Introduction

Finance Field has long been regarded as a prospective field in Machine learning, considering the price of financial assets is always non-linear, dynamic and chaotic, namely, it is difficult to predict.[1] Many famous organizations, including American Accounting Associates(AAA), EMERJ have all developed their own research areas. Models including RNN, LSTM have all proven to be efficient in predicting the future trend of finance grows in stocks, shares and currency flow.

Bitcoin is a special type of virtual currency, it is broadly used in series of online trading systems. In the past decades, the price of Bitcoin has went through series of fluctuation. Nowadays, the average price of Bitcoin is about 7000 from BTC to USD. It is an ideal platform to test the machine learning models as well as traditional time series prediction as its relatively young age and resulting volatility.[2]

ARIMA has been shown to be one of the most commonly used algorithms in time-series data prediction. It is applied to forecast prices and performs satisfied[3]. Comparing to ARMA, it is more precise, and take less time to make calculation. In addition, LSTM have proven to be an efficient tool in making price prediction, risk-recognition due to the temporal nature of bitcoin data.

## 2 Dataset

Bitfinex have provided an API for users to reach the real-time pricing information about Bitcoin, it can automatically collect data from this API and result will show in its homepage. Pycurl is a commonly used python library in collecting online data, which support users to request information communication between server and terminal, suitable for receiving data from Bitfinex website. In order to obtain a trainable, considerable price dataset, data is collected from the API, after extract the information (in JSON format), change the Dataframe format, initial data will finally convert to a suitable dataset for later different scenarios analyzation.

Based on the method described above, by using 5-seconds interval trading data on the website Bitfinex, 10000 prices information, include price between BTC/USD, ETH/BTC, ex cetera is collected. And then it is splinted into training set and testing set, each contains first 8000 and last 2000 values, use price of each 5 second for training model. Table 1 shows some fundamental attributes of the information of BTC/USD.

**Tab.1** dataset attributes

| data | maximum | minimum | mean |
|---|---|---|---|
| training set | 7413.5 | 6801.9 | 7173.6274 |
| testing set | 7275.9 | 7130.0 | 7206.831 |

## 3 ARIMA model

In statistics and econometrics, and particular in time series analysis, an autoregressive integrated moving average (ARIMA) model is a generalization of an autoregressive moving average (ARMA) model. The 'integrated' refers to the number of times need to difference a series in order to achieve stationary, which is required for ARMA models to be valid. In other words, ARMA models is equivalent to an ARIMA model of the same MA and AR orders[4]

In this section, the description of the proposed ARIMA model and the general statistical methodology are presented as follows:

Y3hua@eng.ucsd.edu

### 3.1 Data Pre_requisition

Since ARIMA model is used for forecasting a time series which can be made to be 'stationary', after gaining the dataset. In most of the competitive electricity markets this series presents: high frequency, nonconstant mean and variance, and multiple seasonality.[4] Therefore the stationarity and seasonality of price data should be checked, performing differencing if necessary and choosing model specification ARIMA(p,d,q).

Considering the tiny gap between two neighbor price, an average price in a time pried can be adopted, the plot constructed by results that take an average for every 12 prices, which denoted the average price each minute, then its relative autocorrelation plot is also presented to show the stationarity of dataset.

A stationary series roams around a defined mean and its ACF plots reaches to zero fairly quick, while for the original price series, its ACF is slow-decaying, PACF first bar $\hat{\rho}_{11} \approx 1$, both implying price data is non-stationary.
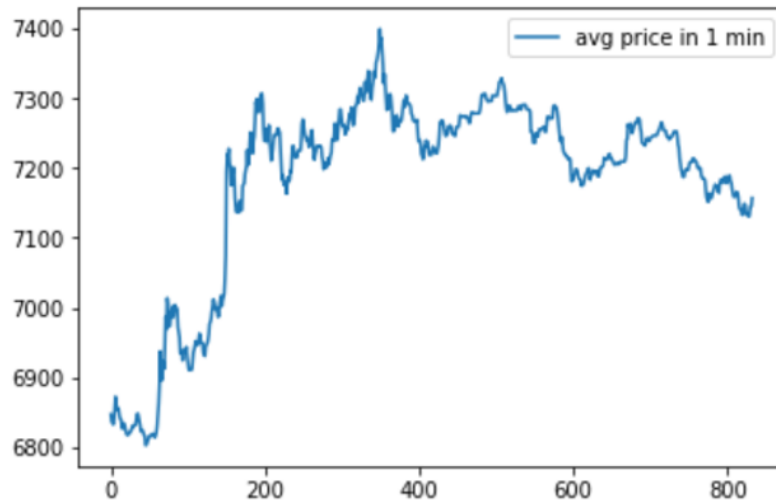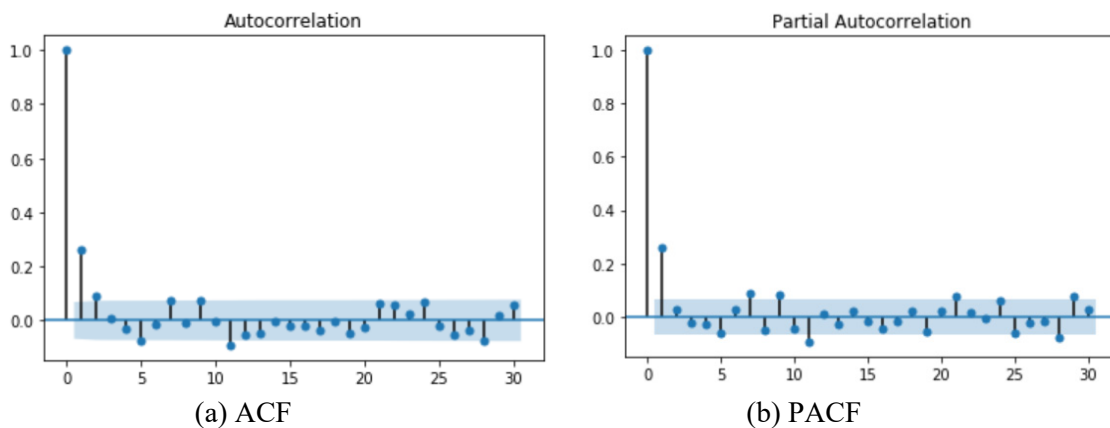


**Fig.1** Averaged price for each minute



(a) ACF          (b) PACF

**Fig.2** ACF and PACF of bitcoin price

### 3.2 Differencing.

As previous result denotes that the initial time series is non-stationary, it is necessary to perform transformations to make it stationary, the most common way is to difference it, the right order of differencing is the minimum differencing required to get the near-stationary series to avoid over-difference. After lag-1 differencing, the result(Fig.2) and correspondent ACF describes its fast-decaying feature, which is a convenient evidence to support the stationarity of differencing data. Simultaneously, both ACF and PACF eventually decayed exponentially to zero, shows refined data is able to construct an ARIMA model.

To avoid high random probability, Lagrange multiplier statistic test for heteroscedasticity. The p-value is 5.8137286e-14, much less than 0.05, ruled out the possibility of white noise.
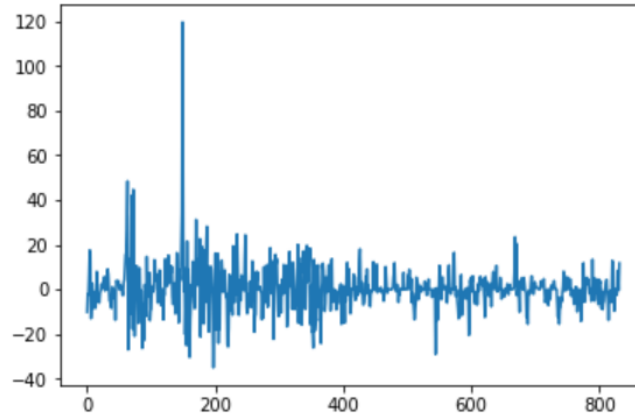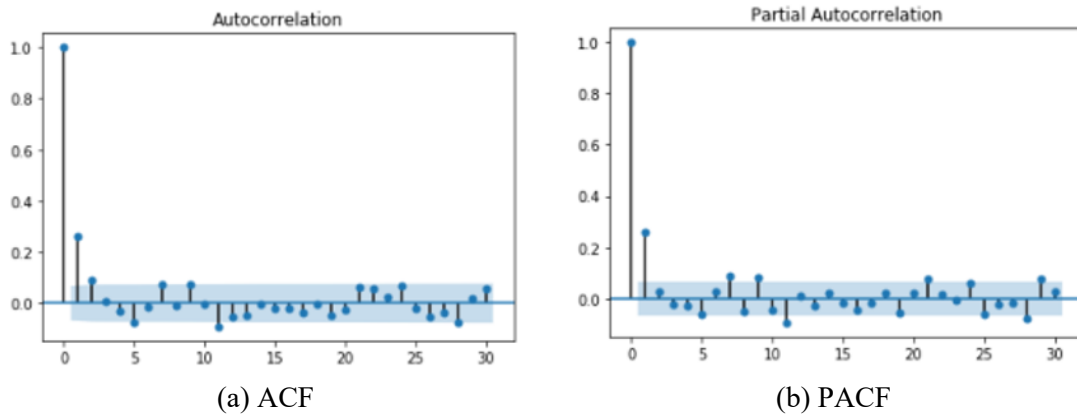
**Fig.3** Price series differencing result



(a) ACF  (b) PACF

**Fig.4** ACF and PACF of 1-leg difference

### 3.3 Forcasting

After data pre-processing, the refined dataset contains 733 points in training set and 100 points in testing set separately. Model is autocorrelated, ARIMA(1,1,0), which is used for differenced first-order autoregressive model, is suitable for forecasting, by regressing the first difference of Y(in this case is the bitcoin price), on itself lagged by one period. The model would yield the following prediction equation:

**Tab.2** p,q values based on ACF and PACF plot

| | AR(p) | MA(q) | ARMA(p,q) |
|---|---|---|---|
| ACF | Tails off | Cuts off after lag q | Tails off |
| PACF | Cuts off after lag p | Tails off | Tails off |

$$\widehat{Y_t} - Y_{t-1} = \mu + \varphi_1(Y_{t-1} - Y_{t-2})$$
$$\widehat{Y_t} - Y_{t-1} = \mu$$
$$\widehat{Y_t} = \mu + Y_{t-1} + \varphi_1(Y_{t-1} - Y_{t-2})$$

Fig.5 presents the prediction of the next 5-step and next single step price based on previous price.



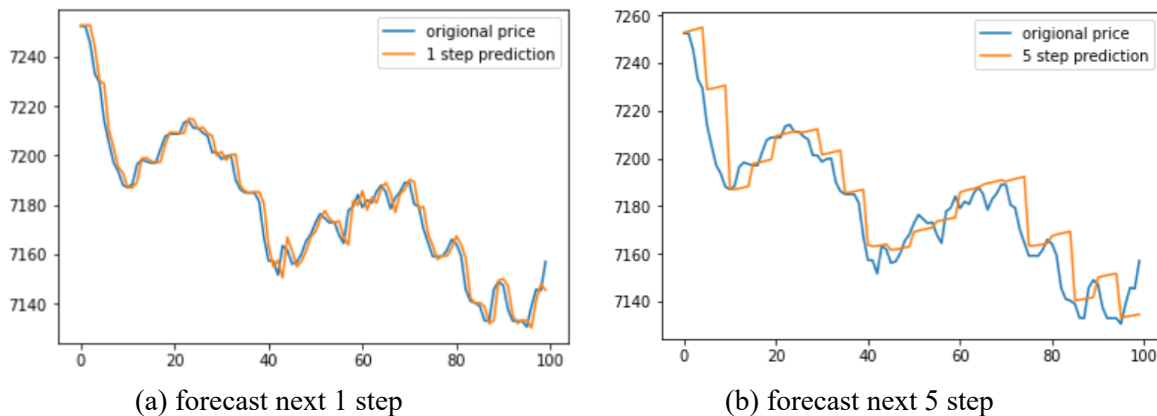(a) forecast next 1 step  (b) forecast next 5 step

**Fig.5** Forecast based on ARIMA

## 4 LSTM model

Long short-term memory (LSTM) is developed from Recurrent neural network (RNN) model to solve the vanishing gradient problem. Comparing to the traditional Front forward neural network (FNN), the RNN adds a self-connecting edge to every node in the network, and thereby allowing the neural network to utilize the prediction result from the last run, to make time-series prediction.[5] This characteristic allows programmers to process various data, including handwriting recognition, speech recognition and anomaly detection.

LSTM could be regarded as an improved version of RNN, it adds the input gate, output gate and forget gate to the existing cell units. These additional gates allow the cell unit to discard useless information, and memorizing important information in the training process. These gates could also handle with the exploding and vanishing gradient problems. In nowadays, LSTM is the most broadly-used tool to making classifying, processing and making predictions based on time series data.
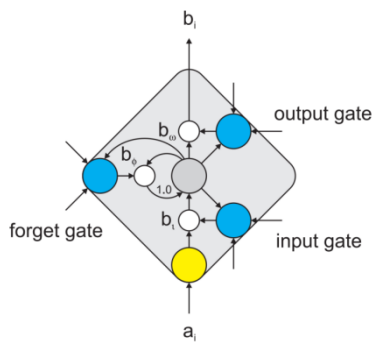


**Fig.6** An example of LSTM cell unit

The forget gate takes the h and x as input, and output a decision (0~1) to the other parts. 0 represents "forget all", whilst 1 represents "keep all". In the input gate, the sigmoid function would decide which information to renew. Finally, the output gate would decide which value to output. It will combine the information from different parts, and decide which information to keep, and which information to output.

The model is trained 100 epochs with 10 pieces of continuous data in each round, the loss plot (fig.7) recorded from each epoch shows that loss initially be large and then convergence immediately. The loss close to 0.02 at first, after 5 epochs, it becomes around $3 \times 10^{-5}$ and reduce its decaying rate. With different input training shape, which is the time length for predicting next step's value, loss growth as time period stretch. Loss based on different strategy of input time series are also presented in the figure.
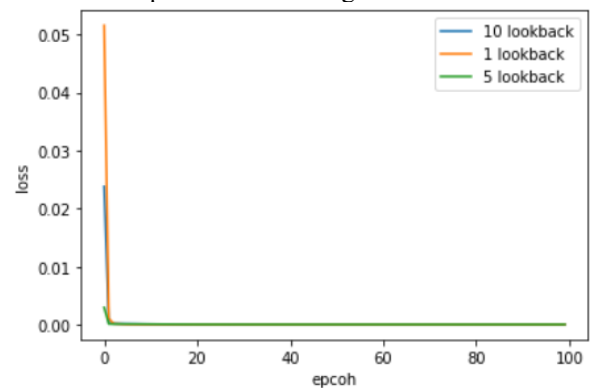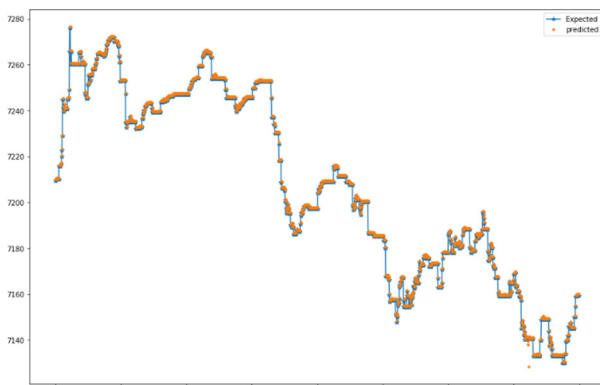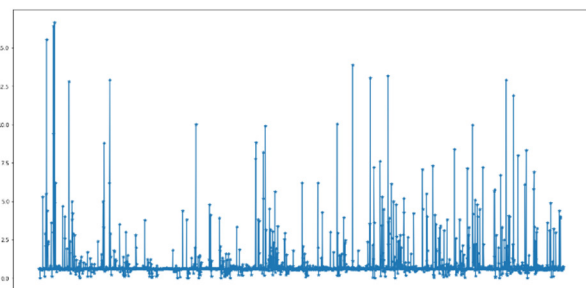


**Fig.7** Training Loss

In the training process, it is obvious that at the beginning of training, the loss is relatively high, but it could decrease sharply as the epoch goes up. After 3~4 epochs, the loss is very close to zero already.



(a)prediction based on 1 previous data



(b) error distribution

**Fig.8** LSTM prediction result

Trained LSTM model performs a satisfying prediction of testing set. The average Error rate is 0.4765938, with a standard deviation of 2.092208. This is ignorable since the original data is quite large. Model that use 5 and 10 previous data is also tried in testing set and comparing to single previous point, 5 or 10 points considered for forecasting is actually having negative effect, even to capture the fluctuation of data, the predicted result is not as precisely as expected.
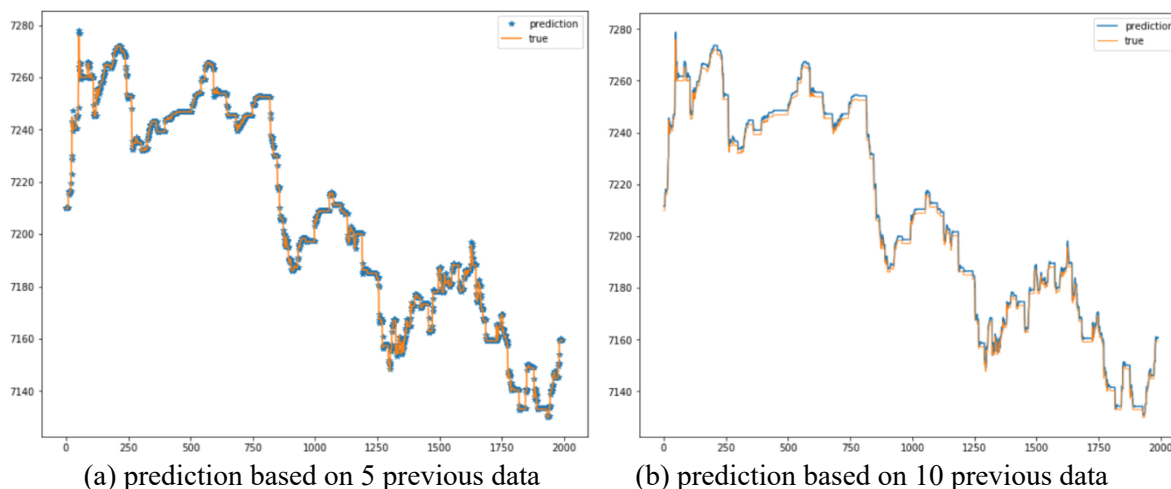
| (a) prediction based on 5 previous data | (b) prediction based on 10 previous data |

**Fig.9** prediction with different strategy

In Dec 14th, another test set, containing captured 3752 pieces of information from the Bitfinex, is also tried for the model. The prediction result shows that LSTM is also performing well in datasets collected from various time periods.

## 5 Conclusion

Although both ARIMA and LSTM could perform well in predicting Bitcoin price, the LSTM would take extra amount of time to train the neural network model for about 42 minute an epoch via 4 core CPU or 1 minute 12 seconds via 2 core GPU. However, after training, the LSTM could make prediction more efficiently, and the precision rate is also higher. In general case, taking less previous data to make prediction in LSTM could lead to better result. ARIMA is quite efficient in making prediction in short span of time; but as the time grows, the precision rate would decrease.

## References

1. B.M. Henrique, V.A. Sobreiro, H. Kimura, Literature review: Machine learning techniques applied to financial market prediction,Expert Systems with Applications, 124 (2019), pp. 226-251.

2. S. McNally, J. Roche and S. Caton, "Predicting the Price of Bitcoin Using Machine Learning," 2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP), Cambridge, 2018, pp. 339-343.

3. E. Weiss, "Forecasting commodity prices using ARIMA", vol. 18, no. 1, 2000, pp. 18-19.

4. J. Contreras, R. Espinola, F. J. Nogales and A. J. Conejo, "ARIMA models to predict next-day electricity prices," in IEEE Transactions on Power Systems, vol. 18, no. 3, 2003, pp. 1014-1020

5. T. Phaladisailoed and T. Numnonda, "Machine Learning Models Comparison for Bitcoin Price Prediction," 2018 10th International Conference on Information Technology and Electrical Engineering (ICITEE), Kuta, 2018, pp. 506-511.