Bitcoin Transaction Strategy Construction Based on Deep Reinforcement Learning

Feng-Rui Liu
Northeast Electric Power University
Jilin city, China
lfr0007@aliyun.com

Ming-Yao Ren
Northeast Electric Power University
Jilin city, China
dongbeidlry@163.com

Ji-Di Zhai
Northeast Electric Power University
Jilin city, China
zjd_1575@163.com

Guo-Qing Sui
Northeast Electric Power University
Jilin city, China
141405334@qq.com

Xiao-Yu Zhang
Central University of Finance and Economics
BeiJing, China
101057797@qq.com

Xiang-Yun Bing
Northeast Electric Power University
Jilin city, China
142371871@qq.com

Yun-Lian Liu*
Northeast Electric Power University
Jilin city, China
158265257@qq.com

***Abstract*—This study proposes a framework for automatic high-frequency transactions based on a deep reinforcement learning model — proximal policy optimization. The framework creatively regards the transaction process as actions and returns as awards to align with the idea of reinforcement learning. It compares price prediction policies including long short-term memory (LSTM) and multi-layer perception (MLP) by applying them to the real-time bitcoin price. Then an automatically-generating transaction strategy is constructed building on PPO with LSTM as the static policy. The approach is able to trade bitcoins in a simulating environment with synchronous data and achieves []% higher benefit than that of the average market. It is demonstrated that the approach can earn excess returns and be expanded to other financial products.**

Keywords—Bitcoin; Deep Reinforcement Learning; Proximal Policy Optimization

I. INTRODUCTION

Bitcoin leverages the blockchain technology to form a wildly-circulated kind of virtual currency. Recently, the application scenarios of virtual currency have ushered in explosive growth, gradually forming a burgeoning investment market.

Previous researchers mainly focused on classic methods borrowed from the financial domain for bitcoin price predictions. Dian employs the α-sutte factor and obtains better performances [1], whose credibility is reduced though when it is proved that bitcoin returns could not be predicted by explanatory variables [2], leading the factor construction methods to fell into a deadlock temporarily. However, traditional approaches mostly can not adapt to such a brand-new situation with violent fluctuations and less technical indexes.

With the development of neural networks, artificial neural network (ANN), neural network auto-regression (NNAR) and LSTM provide more credible methods for long-term time-series predictions. Deep learning algorithms have been widely exploited to explore the law of tendency in bitcoin price. [3] compares the capability of capturing longer range dependencies between LSTM and SVR, in which LSTM is highly advantageous. Nevertheless, most of the previous studies center around static models, which can only advise in the future trend instead of direct decision-making. Therefore, they highly rely on experts, that is, manual effort, leading to being inevitably interfered by subjective factors.

However, bitcoin price fluctuates dramatically, bringing huge challenges to static trading strategies. With the advent of state-of-art techniques such as reinforcement learning, it is expected to take advantage of new-coming technologies to address the challenges posed by the emerging market. Hence, in this study, a novel framework of automatically generating high-frequency transaction strategies is proposed. Advanced deep learning algorithms like MLP and LSTM are employed to predict the price of bitcoin as the base policy. Having back-tested the two policies, LSTM is opted according to its higher accuracy to structure a deep-reinforcement-learning agent for automatic high-frequency transactions based on the PPO algorithm, where the trading actions are creatively regarded as the movement of characters and returns are regarded as scores

978-0-7381-3122-1/21/$31.00 ©2021 IEEE

in a game. Experimental results prove that the automatically constructed policy and the decision-making trading strategy can receive excess returns.

II. Methodology Background

A. Long-Short Term Memory (LSTM)

LSTM plays an pivotal part of the Recurrent Neural Networks (RNN) family, utilizing continuous observations to learn temporal dependencies to predict the future trend. Each LSTM is a set of units that capture the data flow. These units connect from one module to another, transmit data stream past by, and collect the current data flow. There are three kinds of gates responsible for forgetting, inputting and outputting. These gates are organized based on the sigmoid function, borrowed from the neural network layers, enabling the data stream to pass through or be processed selectively by the inner cells.

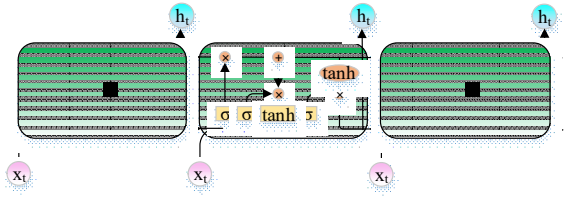

Fig I. LSTM

An LSTM layer maps the input sequence $x = (x1, x2, \ldots, xT)$ to the output sequence $y = (y1, y2, \ldots, yT)$. Specifically, at each time point T_i, there is a corresponding state C_t, which records all previous information, which could be modulated by adjusting the weight input and forgetting, mainly influenced by the above mentioned sigmoid gate, whose way is described by the following equations (1)-(5):

$$i_t = \sigma(W_{ix}x_t + W_{im}m_{t-1} + W_{ic}c_{t-1} + b_i) \quad (1)$$

$$f_t = \sigma(W_{fx}x_t + W_{fm}m_{t-1} + W_{fc}c_{t-1} + b_f) \quad (2)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g(W_{cx}x_t + W_{cm}m_{t-1} + b_c) \quad (3)$$

$$o_t = \sigma(W_{ox}x_t + W_{om}m_{t-1} + W_{oc}c_t + b_o) \quad (4)$$

$$m_t = o_t \odot h(c_t) \quad (5)$$

where W denotes the weight matrix, W_{ic}, W_{fc} and W_{oc} represent the diagonal weight matrix connected by peephole; Indexes i, f and o denote the input gate, forgetting gate, and output gate respectively; b indicates the offset vector and c is the unit activation vector; σ is the sigmoid function; $\odot$ is the product of element direction of vector; g and h are the activation functions of unit input and unit output, usually tanh.

B. Proximal Policy Optimization (PPO)

Proximal policy optimization (PPO) [4] has been followed recently, belonging the policy gradient (PG) algorithm clan. It calculates the estimated straight value of the gradient of a fixed strategy and inserts it into a sort of algorithm named random gradient rising.

The main contribution of PPO is that it leverage the information by combining both the error term of the value function and the policy substitution. Thus, the neural network architecture could be able to share parameters between the policy function and the value function. The objective function is as equation (6) shows:

$$L_t^{CLIP+VF+S}(\theta) = \hat{E}_t[L_t^{CLIP}(\theta) - c_1 L_t^{VF}(\theta) + c_2 S[\pi_\theta](s_t)] \quad (6)$$

where c_1 and c_2 represent parameters; S denotes entropy excitation; L_t^{VF} represents variance loss. Reference [5] proposes an approach implementing the idea of strategy gradient adapted to RNNs. Firstly, it executes the strategy with t time steps, where t is far less than the episode length. The learning strategy is also updated by the collected samples. The advantage estimator as equation (7) shows is in dispensable for this approach.

$$\hat{A}_t = -V(s_t) + r_t + \gamma r_{t+1} + \ldots\ldots + \gamma^{T-t+1} r_{T-1} + \gamma^{T-t} V(s_T) \quad (7)$$

where t is a time point inside the rage of [0, T]; γ is the incentive discount rate. When λ=1, it can be rewritten as follow:

$$\hat{A}_t = \delta_t + (\gamma\lambda)\delta_{t+1} + \ldots + (\gamma\lambda)^{T-t+1}\delta_{T-1} \quad (8)$$
$$\text{where} \quad \delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$$

[4] lists specific derivation processes of the above equations and provides mathematical proof. It also specifically explains how to standardize the above equation (8) in virtue of generalized advantage estimation.

III. The Proposed Method

A. Selecting a Static Deep Learning Policy

Referring to previous related research, this study compares the performances of predicting static data. Following the traditional MLP defined by predecessors, the network structure is set to three layers, namely hidden layer, output layer and input layer.

The structure of the LSTM network is constructed by 4 LSTM layers on the top of a fully-connected layer. The first layer is responsible for receiving input and the rest LSTM layers are hidden layers. The last layer controls output. Details includes:

- Activation function: Relu for LSTM layers; linear for the dense layer
- Optimization algorithm: Adam
- Loss function: Mean Square Error (MSE)
- Evaluation indexes: RMSE, MAE and R-square

Hyper-parameters are decided through the grid method. The best performance's corresponding hyper-parameter combination is {batch size = 32, hidden units = 50, training epochs = 300}. The best time step for LSTM is 40, and for MLP is 20.

TABLE I. Performances of LSTM and MLP

Models	Indexes of Static Prediction		
	MAE	***RMSE***	***R-square***
LSTM	39.9384	86.6926	0.9962
MLP	177.8798	195.4166	0.9808

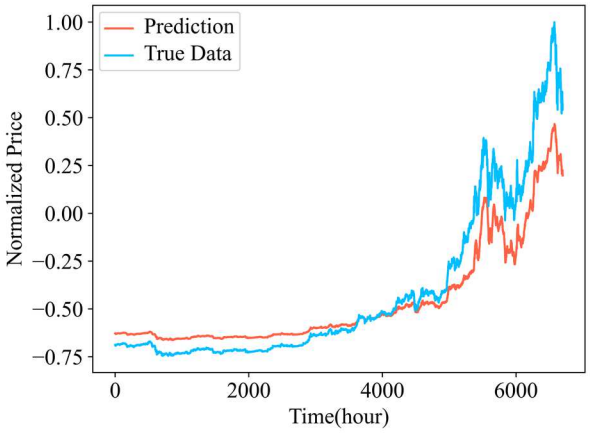

Fig II. MLP Prediction Results

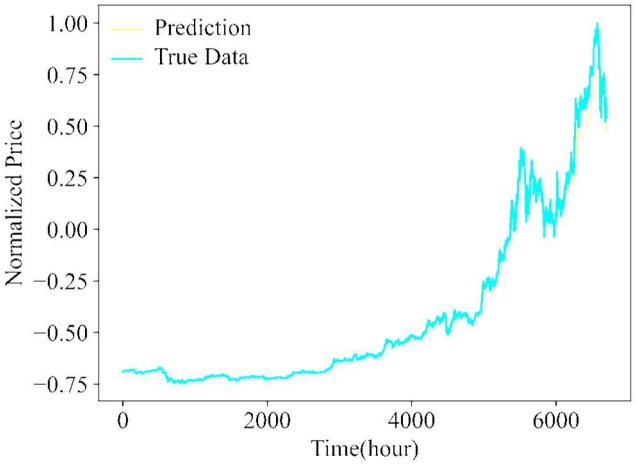

Fig III. LSTM Prediction Results

The predicted price is close to the actual value, but when the bitcoin price fluctuates violently, there is still an obvious divergence on the whole. This is partly because the historical close price information can not fully reflect all information of the trading market. Fig. II and Fig. III show the results of the off-line experiment.

B. *PPO Agent Construction*

The simulation environment is Gym, supplied by OpenAI. The agent is initially assigned $10000 as the start-up capital; the transaction fee rate is 0.25%; the minimum trading unit is 0.125 bitcoin; the action space cover three types of actions (i.e. "buy," "hold" and "sell") at each time point, thus, 24 actions are in the action space.

80% of the data set is split as the training set and the rest 20% is the test set. According to previous experimental results, this study chooses LSTM as the static policy. Omega Ratio is selected as the reward indicator shown in (9).

$$\omega \triangleq \frac{\int_r^{\infty}(1-F(x))\,dx}{\int_{-\infty}^{r}F(x)\,dx} \qquad (9)$$

where r denotes the specified critical rate of return; $f(x)$ is the cumulative distribution function of the yield.

C. *Bayesian Optimization*

Bayesian optimization is a technique for effectively searching the hyperspace to discover the best hyper-parameter combination to optimize the objective function. It assumes the candidate space as compact or discrete and thus transform the parameter-tuning problem into a sequential decision-making problem. As the iteration progresses, the algorithm continuously observes the relationship between the parameter combination and the objective function value. It opted the observation aim by optimizing the acquistion function, which balances the unexplored points and the best value of explored points. It also introduces regret bound to achieve state-of-art effects.

D. *Visualization*

The results are visualized to show the process of trading on the test set by the trained agent to enlighten trading programmers. The user-friendly interface is shown in Figure IV., which is dynamic while trading.

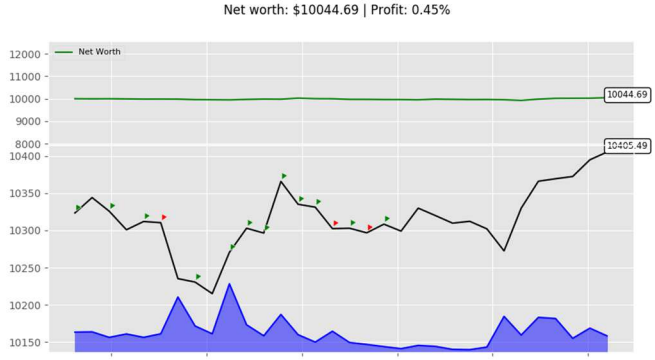

Fig IV. Interface for the Trading Agent

The red dot indicates the buying point while the greed dot indicates the point of selling; net worth shows the current assets and profit is the yield rate; the black line indicates the tendency of bitcoin price; the green line indicates the trend of returns; the blue area represents the volume of bitcoin.

E. *The Flow of Proposed framework*

The proposed framework is as follows:

- Initialize a Gym trading environment.
- Setup the framework and trading sessions.
- Decide the static policy, the award function, and the optimization tool.
- Train the agent and evaluate it on the test data set
- Visualize the final dynamic trading process.

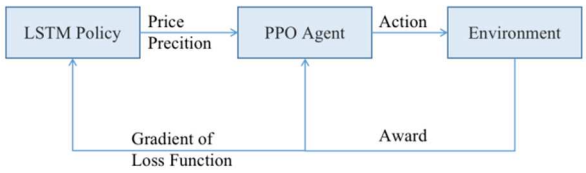

Fig V. Flowchart of Proposed framework

IV. EXPERIMENTS

A. Data Set

In this study, the data set comes from the website cryptodatadownload[1]. There are 30984 valid records, covering the time period from 4:00 a.m. on Aug 17th, 2017 to 0:00 a.m. on Feb 27th, 2020.

Following the previous setting, this data set is divided into training set and test set according to 8:2. In order to enhance the convergence speed, it's necessary to summarize the statistical distribution of a uniform sample. This study adopts the minimum-maximum value normalization approach to normalize data, whose equation is shown in (10):

$$origin_i^* = (y_{max} - ymin) \times \frac{origin_i - origin_{min}}{origin_{max} - origin_{min}} + y_{min} \quad (10)$$

where $origin_{max}$ and $origin_{min}$ represent the maximum and minimum values of the original data respectively; *y* denotes the processed data.

B. Benchmarks

Benchmarks are chosen from technical strategies, including the Buy and Hold, the Chasing Up and Killing Down strategy and the Momentum strategy. [19]

(i) Buy and Hold: buys BTC at time t=0 with the initial capital and sell it only once at the end of the total period when profits are evaluated.

(ii) The Chasing Up and Killing Down strategy: (1) If at time *t,* the average increase from *t-5* to *t* is higher than the average increase at *t-20* to *t* by *r*% ($r > r_0$), then buy $r \times u$ bitcoins, where r_0 and *u* are preset quantities. r_0 is usually defined as 5 and *u* is generally 0.0125; (2) If at time *t,* the average decline from *t-20* to *t* is higher than the average decline from *t-5* to *t* by *r*% and $r > r_0$, then sell $r \times u$ bitcoins.

(iii) The Momentum strategy: assumes the price changes at time *t+1* will be the same as at time t, and make trading decisions according to the prediction.

C. Experiment Results

The proposed model has been significantly improved the performances of baseline and the yield on the test set is 14.74% higher than that of the best benchmark. Details are shown in Table II.

TABLE II. PROFITS RATE OF DIFFERENT STRATEGIES

Strategies	Profits Rate / %
Proposed Framework	1.46
Buy and Hold	- 39.09
Chasing Up and Killing Down	-26.23
Momentum	-13.28%

Compared with the baselines, the proposed strategy has obvious advantages. It gets 14.74% more profit than the Momentum strategy, 27.69% more profit than the Chasing Up and Killing Down strategy and 40.85% more profit than the Buy and Hold strategy. Besides, it can earn excess returns when the financial market including the bitcoin market is a slump influenced by the COVID-19 pandemic.

The proposed model succeeds to achieve profitability, though it still has space for improvement. The agent may not be able to allocate assets evenly, resulting in excessive purchases of bitcoin in the early stage, leading to the lack of cash after that. Therefore, it may not be able to seize the opportunity to increase positions.

V. CONCLUSION

This study demonstrates the prediction effect on the bitcoin price of various types of deep neural networks, and proves the possibility of letting an trained agent to automatically trade bitcoins and earns excess profit based on deep reinforcement learning.

It proposes a general framework for automatically generating a high-frequency trading strategy with a PPO-based agent. Moreover, the agent is proved to have better performances than traditionally popular strategies and be able to earn high returns even when the market is a slump and the price fluctuates. Excitingly, the proposed framework could easily extend to other financial transactions, such as stocks, features and other digital currency such as Doge coin and Ethereum (ETH).

The extensive application scenarios of cutting-edge algorithms outside the field of games have been proved by this study. The method is proposed to make up for the lack of traditional quantitative strategies in high-frequency trading Additionally, it highlights potential influence of factors in the trading simulation, which will enlighten researchers and investors.

REFERENCES

[1] Sutiksno, Dian Utami et al. "Forecasting Historical Data of Bitcoin using ARIMA and α-Sutte Indicator.", 2018.

[2] Aalborg, Halvor Aarhus, Peter Molnar, and Jon Erik de Vries. What can explain the price, volatility and price, volatility, and trading volume of Bitcoin?. Finance Research Letters, 2018.

[3] Basheer I A, Hajmeer M N.Artificial neural networks: fundamentals, computing, design, and application[J].Journal of Microbiological Methods, 43(1):3-31, 2000.

[4] Schulman J, Wolski F, Dhariwal P, et al. Proximal policy optimization algorithms. arXiv preprint arXiv:1707. 06347, 2017.

[5] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In: arXiv preprint arXiv:1602.01783. 2016.

[1] https://www.cryptodatadownload.com/