MBAN 6510
Artificial Intelligence II

Trading via Reinforcement Learning

by Team 6:
Aniruddh Kumar (217107798)
Evgeni Averkin (210772929)
Nikita Khaliapin (217470709)
Xiao Quan (216917207)

Introduction

The purpose of this report is to describe how a reinforcement learning (RL) algorithm was combined with a Bitcoin prediction model to create a profitable trading strategy. The prediction model combines a Long short-term memory (LSTM) network architecture with an additional data source in the form of a Crypto Fear & Greed Index (CFGI) to predict Bitcoin's daily close price. The model's LSTM component utilizes data from the last ten days along with the previous day's CFGI to predict Bitcoin's next day close price. The model's price predictions are then used as the training data for the RL agent. The goal here is to create a trading strategy where the agent makes the most profitable trades while accounting for future predicted Bitcoin prices. The most optimal trading strategy would be that which will achieve the largest profit on a simulation that started with a balance of 10,000 USD and 10 BTC and traded during a period from Feb 1, 2018, to Oct 23, 2019.

Methods and Results

The Bitcoin prediction model is a linear model that consists of two inputs from separate LSTM models. The first LSTM model preprocesses Bitcoin's historical daily price by re-indexing, passing only the closing price to the model, and then predicting the closing price based on the past ten days. The second LSTM model collects CFGI data through an API and then uses the last day's CFGI and Bitcoin price data to make another prediction on the closing price. Then the two predictions are joined together into a linear model to make the final prediction. The linear model achieves a prediction model with an MSE of 118,402 for the period of Feb 1, 2018, to Oct 23, 2019.

The next step is to pass the set of predictions from the linear model as the training set for TensorTrade, which is a framework that generates a trading strategy using reinforcement learning. A trading environment was set up by feeding the daily closing bitcoin price in USD to the pool. The environment uses 'managed-risk' as the action scheme, which means that actions are based on managing risk by setting a follow-up stop loss and taking profit on every order[1]. Bitcoin trading is very volatile, and so such an action scheme will create an optimal risk management strategy that will shape odds in the trader's favor. The environment's reward scheme is 'risk-adjusted', which means it rewards the agent for increasing its net worth while penalizing more volatile strategies[2]. An Advantage Actor-Critic (A2C) agent was trained on the predicted price in this environment. The A2C agent has an actor-critic feedback system that can perform well in a trading environment. At the training's start, the agent does not know how to trade, and so it tries some action randomly. The critic observes the action and provides feedback. The agent then learns from the feedback, updates its policies, and improves its trading actions. Simultaneously, the critic updates its way of giving feedback. The A2C agent uses the past four days of Bitcoin price close predictions to form its trading actions and overall strategy.

After training, the A2C agent was backtested on a period from Feb 2, 2018, to Oct 23, 2019. This period represents 628 days, where the agent makes a trading decision every single day. The final result is a net worth increase from 98,388USD to 108,726USD (**Figure 1**). Furthermore, the agent produced a positive mean reward of 702,996, which means that it has undertaken more profitable trades than negative ones.

[1] https://github.com/tensortrade-org/tensortrade/blob/master/tensortrade/actions/managed_risk_orders.py
[2] https://www.tensortrade.org/en/latest/components/rewards/RiskAdjustedReturns.html

Discussion

Although the preliminary results show a positive return, the agent does not guarantee to achieve such a result every time. Subsequent training showcases that the agent delivers a net-worth that ranges between $80,000 to $110,000. Such a level of randomness in training must be tuned in the future for more consistent results. Furthermore, the net-worth is highly correlated to the Bitcoin price (**Figure 2**). This indicates that the fluctuations in Bitcoin price have a more significant effect on one's net-worth than trading decisions. Thus, the agent is not able to effectively take advantage of Bitcoin's price volatility.

The model could be improved in the future by adding additional input data in the form of technical indicators such as trend, momentum, volatility, and volume. The model can then integrate them with the reinforcement learning agent for a more efficient generation of stock trading decisions. Technical indicators can also help set up a better action scheme. Currently, the bot is using a managed-risk strategy where a fixed percentage is set for stop-loss and take-profit. Bitcoin's volatility is not a flaw, but a lucrative feature and such a conservative strategy defeat the whole purpose of trading bitcoin. Technical indicators can give signals when to have a more conservative or aggressive approach, and the designer of the system should incorporate them into the action scheme.

The reward policy is also very simplistic and does not represent the trading game. In the model, the system rewarded the agent when net-worth increased. However, the reward needs to be more flexible and reflect market conditions and be able to reward the agent on a per-order basis. For example, the reward needs to take into account the order size, order timeframe, order direction, order duration, etc. Only then will the agent be able to learn the optimal policy. For every market candle, there is an optimal position that the agent can take, and a more robust reward system is required for the agent to differentiate between its actions.

Bitcoin price reflects non-stationary time series data. In principle, it is not necessary to reinforce stationarity for LSTM models, and so our methodology kept the data as it is. However, stationarity could make prediction tasks much more efficient, and stable. Moreover, taking out the actual price values could give the agent a general learning power that would be irrelative to price levels. Such behavior reflects real traders who do not adjust their trading strategy based on price magnitude. Currently, the agent ties specific actions to certain price levels. Instead, you would want the agent to trade Bitcoin the same way at different price levels, and to do so, it is necessary to drop price levels to form a constant picture for the model.[3]

Finally, despite TensorTrade's great features and algorithms, it is still a very raw product with limited documentation and many examples and tutorials that do not work correctly. This lack of support leads to inconsistencies between documentation and actual functions, which limits a novice user from taking advantage of the full capabilities of the library. TensorTrade has a Discord channel with a community where people discuss the library, and using such a resource along with more meticulous research could help strengthen the results in this report.

[3] See TensorTrade Discord post by *MetalMind*, dated 01/25/2020.

Appendix

Figure 1

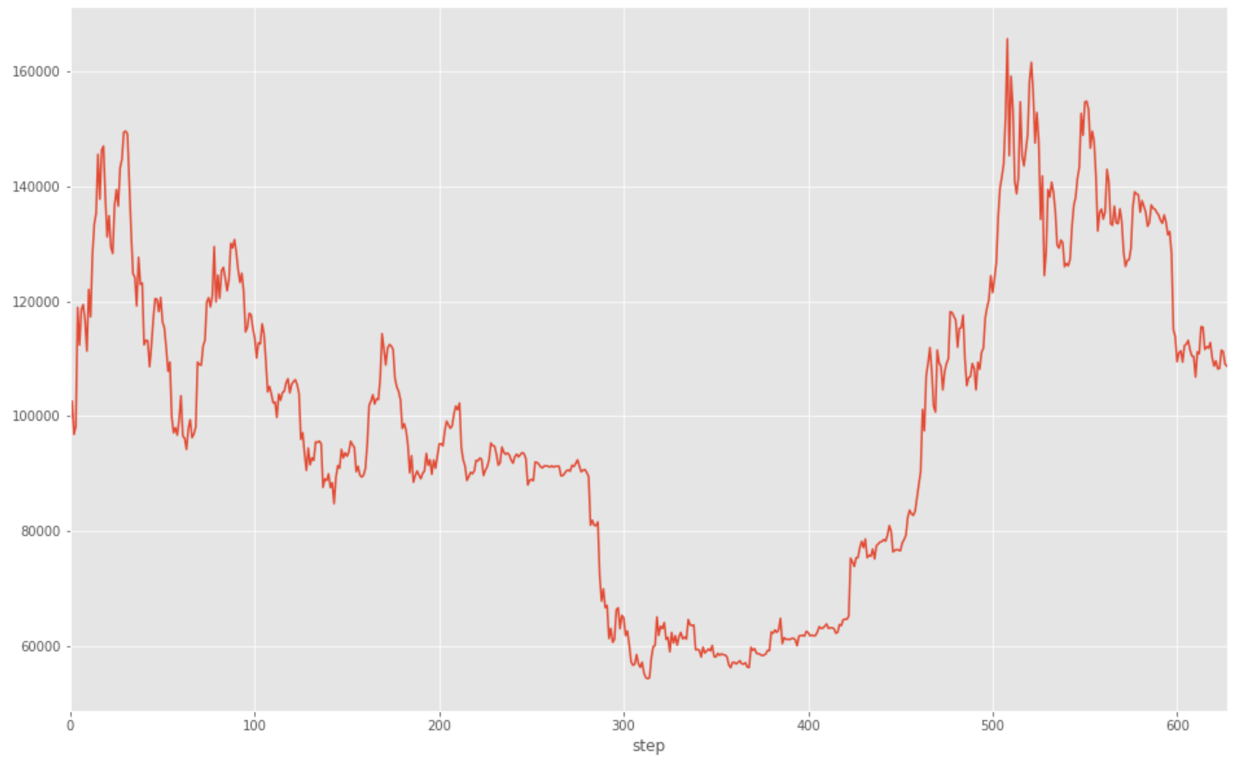
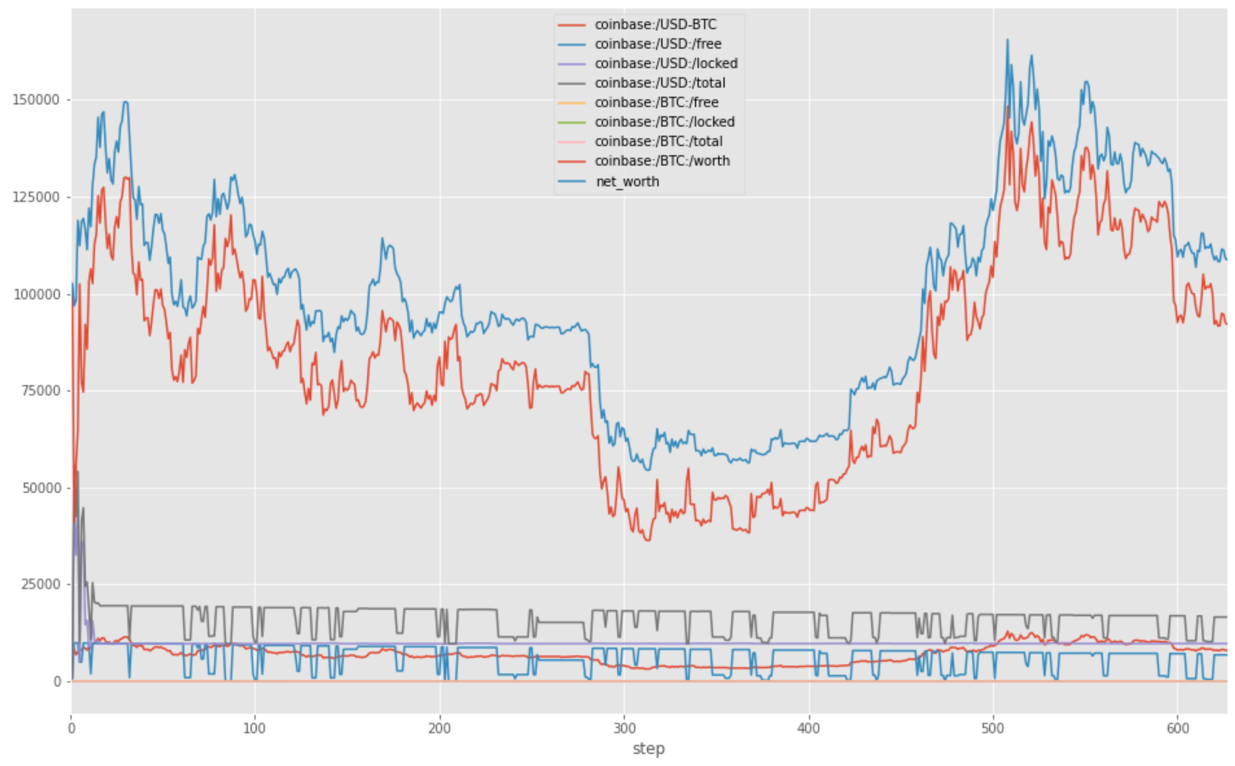


Figure 2


coinbase:/USD-BTC
coinbase:/USD:/free
coinbase:/USD:/locked
coinbase:/USD:/total
coinbase:/BTC:/free
coinbase:/BTC:/locked
coinbase:/BTC:/total
coinbase:/BTC:/worth
net_worth
step