Agent Based Learning for Trading

Yahya JANBOUBI, Ghali LARAQUI

Quick Description

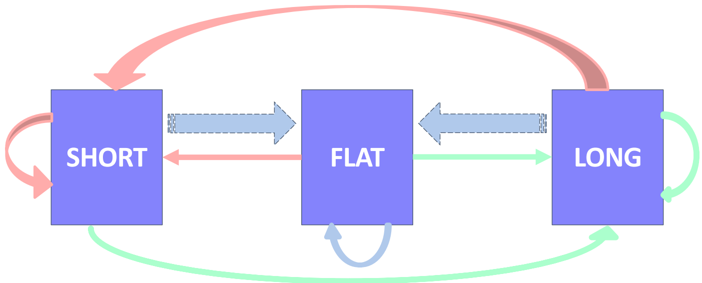
This project is an attempt to develop a low to medium frequency algorithmic trading bot that is trained under a RL paradigm.

The dataset collected relates to prices of the main cryptocurrency in the market, Bitcoin, valued in US dollars. The raw data file comprises minute-by-minute Bitcoin prices from 2015 to April 2021. The number of features in the dataset is the classic ohvc format ('open', 'high', 'low', 'close'), as well as some volume indicators.

As trading has grown more algorithmic, simulated agents offer an interesting approach to try to tackle market movements to derive profitable robust strategies. The recent Dueling DQN architecture (Wang et al., 2016) represents a significant improvement in value-based deep RL, as it allows, through separating value-states and each action's advantage, more precise action value estimation. The key, however, lies in how one leverages the informative power of alternative data.

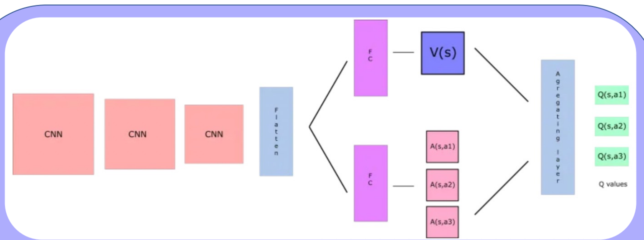
As financial instruments are correlated with one another, minding some lags, and their dynamics, aside from fundamental factors, are driven by socio-psychological factors, it is in the variety and representativeness of alternative features where a model can find more reliable optimal policies. This indeed is a significant limitation of this project for implementation.

Action Space Logic



	ProMax Space	Pro Space
🐻📉	*SELL, DOUBLE_SELL*	*SELL*
🚀🌕	*BUY, DOUBLE_BUY COMBO_BUY*	*BUY*
💎✋	*HODL*	*HODL*

Dueling Architecture Design



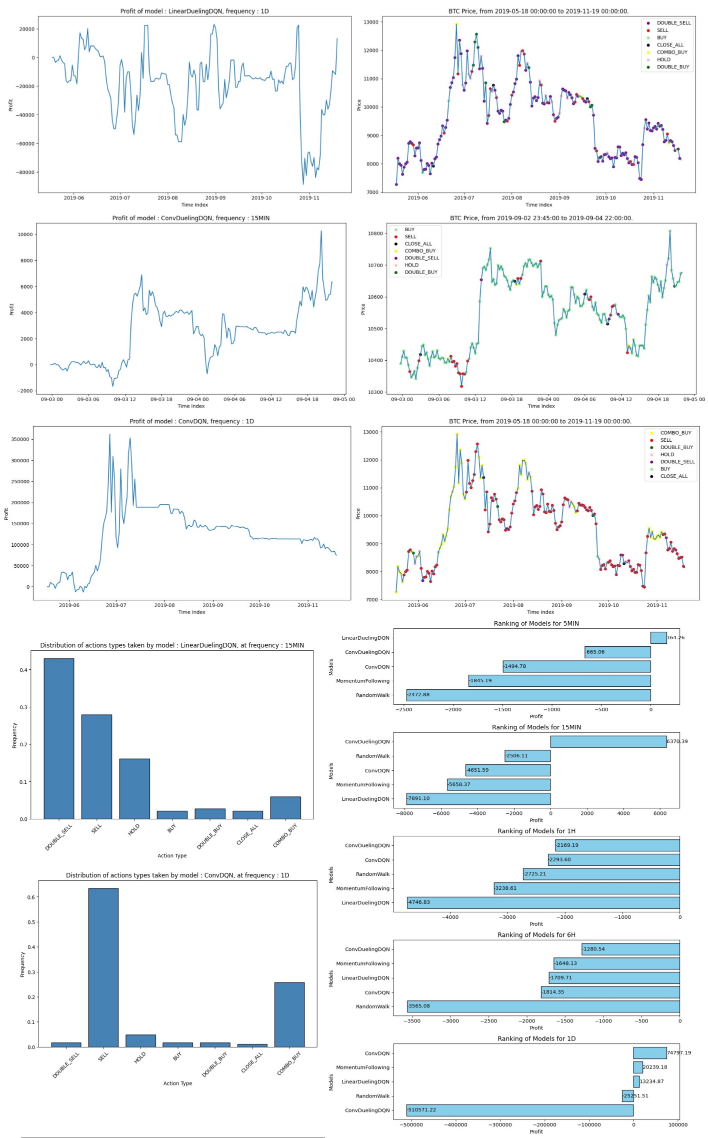
ConvDDQN consists of : the same neural network that splits its last layer in two parts, one of them to estimate the state value function for state s (V(s)) and the other one to estimate the advantage function for each action a (A(s, a)), and at the end it combines both parts into a single output, which will estimate the Q-values.

$$Q(s,a;\theta,\alpha,\beta) = V(s;\theta,\beta) + \left(A(s,a;\theta,\alpha) - \frac{1}{|\mathcal{A}|}\sum_{a'} A(s,a;\theta,\alpha)\right)$$

Structure & Methodology

- Two action spaces of different complexity.
- No limits on the inventory size of an agent's positions. Downside is no real risk management. Fees and market impact were incorporated in PnL computation.
- Various time aggregations were used to cross check the performance of our models : 5 minutes, 15 minutes, 1 hour, 6 hours, and daily granularity.
- 5 different models designed : RandomWalk, MomentumFollowing, Convolution DQN, Dueling DQN (Linear), Dueling DQN (Convolutional).
- Reward mechanism : +10 if PnL > 0 ; -10 if PnL < 0. Customization is terms of aggressiveness (e.g. if HODL when position is FLAT, Reward –= 5).
- A 'Replay Memory' element to enrich the current state (as no alternative data/features were used) and provide more context to the numerical agent.

Results & Discussion



Results are both promising and disappointing. Testing our model's performance over various time frames has been insightful, revealing that profitability increases as data granularity decreases due to reduced noise and clearer price signals. However, at high granularity, actions appear random or biased towards a single type (e.g., sell), likely due to our epsilon-greedy mechanism reducing exploration, which is problematic in volatile markets. Our framework, enhanced by adding alternative data and technical indicators, has the potential to be profitable.