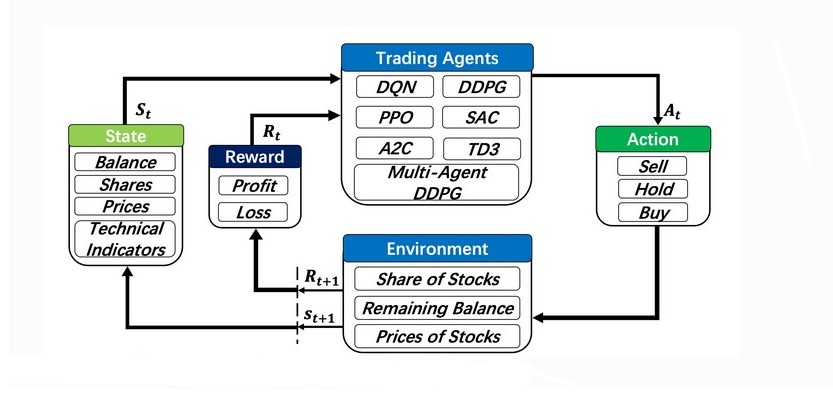
Why did you choose the approach you submitted?

Deep Reinforcement Learning (DRL) is a semi-supervised learning technique and learns by trial-and-error method. In this approach, the agent learns to accumulate reward through a feedback mechanism. It is incentivized through rewards for making sound decisions and incurs penalties for erroneous choices.  Agent (here Crypto-Trader Bot) learns by taking actions (buy, sell, hold) in the environment (crypto market) and receiving fitting reward for profit or loss. Crypto-Trader Bot's goal is to maximize the reward, which in-turn maximises profit.  It works well in case of the unpredictable nature of markets, understanding complex relationships, and finding the best strategies for trading. RL can adapt in real-time, keep learning, and tailor its approach to specific financial goals, like managing portfolios and minimizing risks. It's handy because it can discover new strategies and avoid human biases in trading.

1. Single algorithm agents

We implemented single algorithm agents, primarily because of the simplicity and granular control over various hyperparameters affecting the behaviour of the agent. While training we obtained different insights about how different agents performed under different market conditions. This knowledge can then be used to create an agent which uses different algorithms simultaneously to overcome shortcomings of each of the single algorithm agents.



1. Multiple algorithm agent

Multiple algorithm agent is expected to perform significantly better than each of the single algorithm agents across all the benchmarks of stability like Sharpe ratio, Sortino ratio and max drawdown. The returns should be more consistent and stable as compared to single agent models.

How did you use it?

1. Structure of Code:
2. State Space: A state represents agent’s perception of a market environment. This includes data like Open, High, Low, Close(in form of price array), the indicators used and current holdings of cash and assets.
3. Action Space: This represents the set of possible actions the agent can take at any given instant. Here, in the context of trading of a single asset class, the action space comprises of a continuous range of numbers from -1 to 1, where any number less than -0.1 corresponds to SELL action, any number greater than 0.1 corresponds to BUY action, and any number between -∆ and +∆ corresponds to HOLD.
4. Environment: The crypto trading environment is like a playground for testing and creating computer programs that automatically trade cryptocurrencies. It's carefully set up to mimic real-world trading situations, helping us train and check how well these computer algorithms, or strategies, perform. It's basically a simulation that allows us to see if our trading programs can handle the complexities of the actual cryptocurrency market. This can be modified to suit our requirements.
5. Agent: The agent’s actions are decided by its policy. A policy can be seen as the brain of the agent which can be defined as a mapping from state to action. Here we implement agent’s policy as deep neural networks to account for its complex nonlinear structure.
6. Optimisation: We experimented on various reinforcement learning algorithms like PPO, SAC, TD3, DDPG, A2C to find the best suited for our case. And further we tuned these algorithms by tweaking their hyperparameters.   Some of the hyperparameters include break\_step, gamma and learning rate. We have included an optimisation framework that optimises the model to obtain the best possible value for anu of the required parameters or objectives.
7. Risk Management: We tested two methods for handling risk while maximising the profit. During back testing phase we put a stop-loss measure to handle loss. In our experiments, we put a stop-loss of 1.5% on portfolio value. During our hyperparameter training phase we aimed for a model which returns the highest sharpe ratio metric.
8. Training:

All the agents have been trained on one-day interval data from 01-01-2018 to 31-12-2022 on the asset class (BTC/USDT). Training for shorter intervals yielded poor results on 10-hour training periods, which implies that these models take a lot of time to train for shorter time intervals. Brokerage was also significantly higher on shooter time intervals.

1. Testing:

All the agents have been tested on 3 different time-frames:

• 01/01/2018 to 31/12/2022 - To ensure that our model does not overfit.

• 01/01/2023 to 30/06/2023 - To analyse its performance on an unknown dataset.

• 01/02/2021 to 01/04/2022 - To analyse its resilience in peak volatility.

What did you test before it?

* As the conventional machine learning works on neural networks
* Deep learning models
* RL allows for a balance between exploration (trying new actions) and exploitation (choosing actions that are known to yield rewards), which is crucial in dynamic environments which is not efficient in conventional machine learning model.
* RL is well-suited for problems where decisions are made sequentially over time, such as in trading