REPORT

CSE 4/546: Reinforcement Learning, Spring 2024

Final Course Project

The Environment :

States

The state describes the current market conditions and whether the agent owns any stocks. It's determined by two main factors: whether the average closing price of the stock has gone up over a certain number of days (`price_increase`), and whether the agent currently holds any stocks (`stock_held`). This leads to four possible states:

1. **Price Went Up, No Stocks Held**
2. **Price Went Up, Stocks Held**
3. **Price Did Not Go Up, No Stocks Held**
4. **Price Did Not Go Up, Stocks Held**

Actions

The agent can choose from three actions:

- **Buy (0):** The agent buys as many shares as possible with their available money at the next day's opening price.
- **Sell (1):** The agent sells all their shares at the next day's opening price, turning them back into cash.
- **Hold (2):** The agent does nothing, keeping any shares and cash as they are.

Rewards

Rewards are given based on the financial outcome of the agent’s actions:

- **Buying:** A small positive reward is given if the agent buys shares successfully; there's a penalty if the agent tries to buy without enough money.
- **Selling:** The reward is based on the profit or loss percentage from selling the shares compared to their purchase price (book value).
- **Holding:** Rewards or penalties are based on the increase or decrease in the value of the held shares compared to their purchase price, showing unrealized gains or losses.

Termination

An episode ends when the agent reaches the maximum number of timesteps, which is set by how much data is available minus the number of days used to assess the state. This simulates a trading period ending when the data runs out.

Truncation

The environment doesn't have a specific condition for ending an episode early outside of the typical rules of the game (MDP). Truncation usually happens when an episode ends due to a special condition specific to the environment, but here, it simply ends when there are no more timesteps left.

This environment offers a structured way to test and measure different trading strategies within the typical challenges and decisions of stock trading.

Baseline Algorithms :

Dueling DQN (Deep Q-Network)

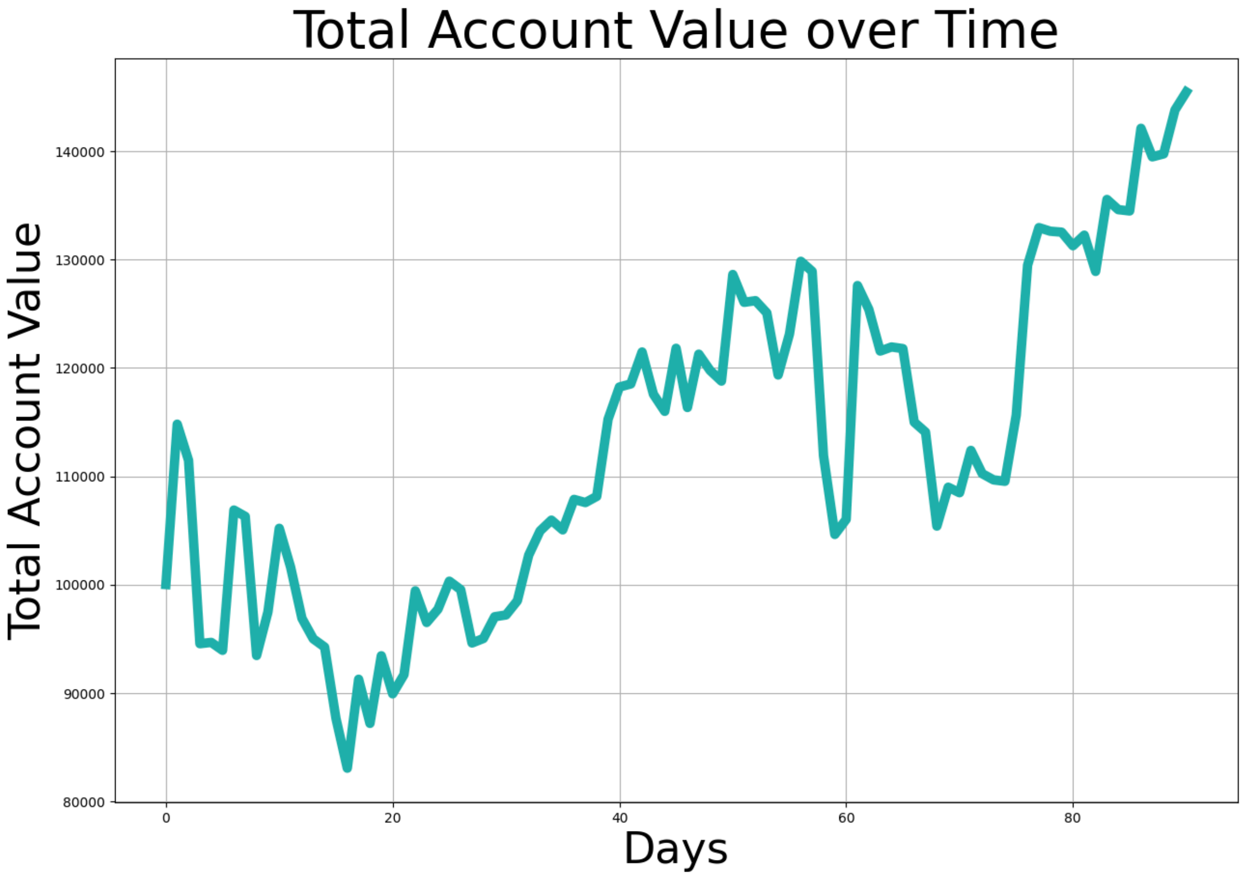
Dueling DQN improves upon traditional DQN by splitting the estimation into two parts: the state value and the advantage of each action. This allows the network to learn about the importance of states independently from the actions taken, which is especially useful when actions do not drastically alter outcomes.

Reason for Choosing Dueling DQN for the Stock Trading Environment

Dueling DQN was chosen for its ability to discern the importance of different states from the actions performed in those states. This feature is vital in stock trading, where the market conditions often dictate the significance of actions. By focusing on state values, Dueling DQN aids in making more informed decisions in critical situations.

Results Using Dueling DQN in the Stock Trading Environment

The "Total Account Value over Time" graph for the Dueling DQN agent shows significant ups and downs with an overall positive trend, indicating effective strategy adaptation to market conditions. The rise in account value demonstrates that Dueling DQN's specialized architecture has successfully managed to enhance decision-making, leading to overall profitability.



A2C or Advantage Actor Critic

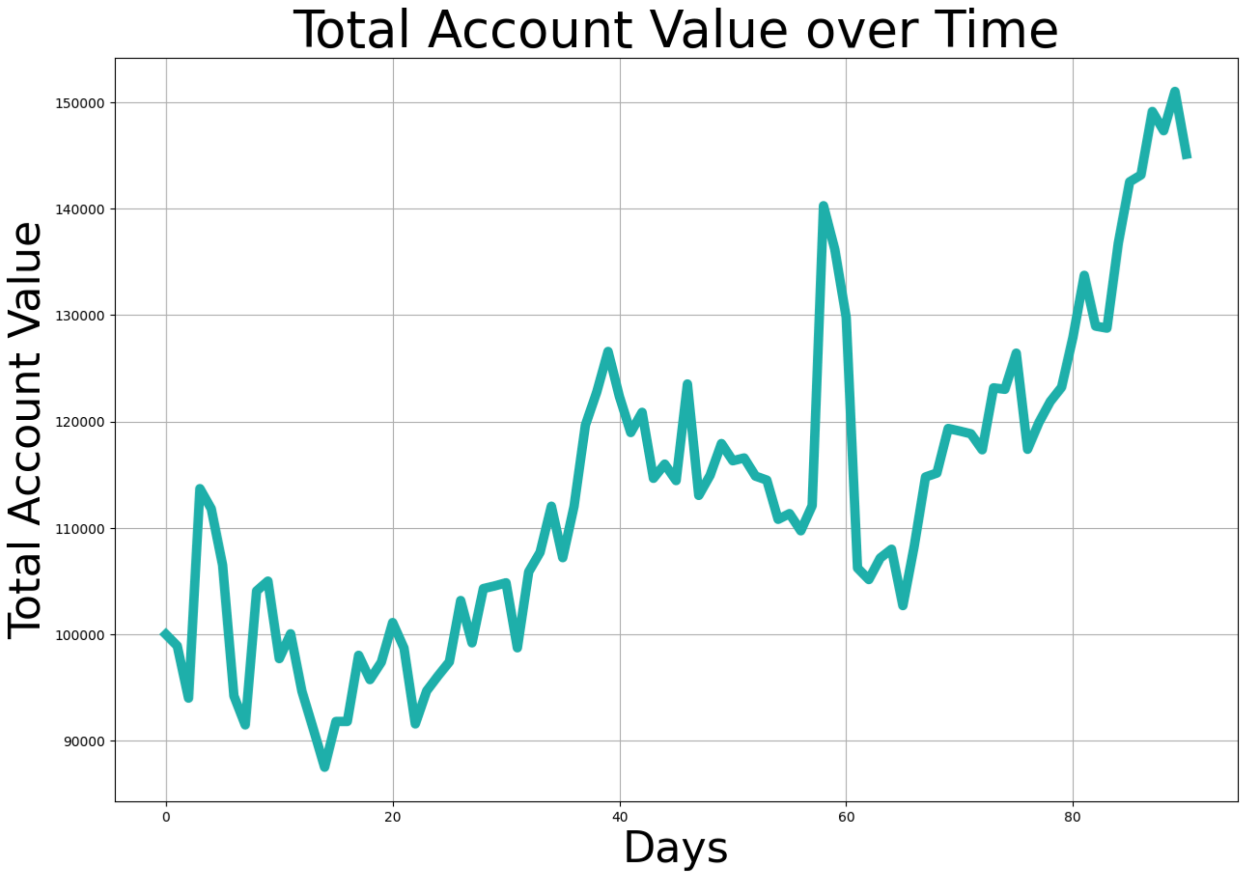
A2C, or Advantage Actor-Critic, combines policy-based and value-based methods in reinforcement learning. The Actor improves the policy based on recommendations from the Critic, which evaluates actions using a value function. A2C efficiently balances exploring new strategies and exploiting successful ones, facilitating faster convergence to optimal solutions.

Reason for Choosing A2C for the Stock Trading Environment

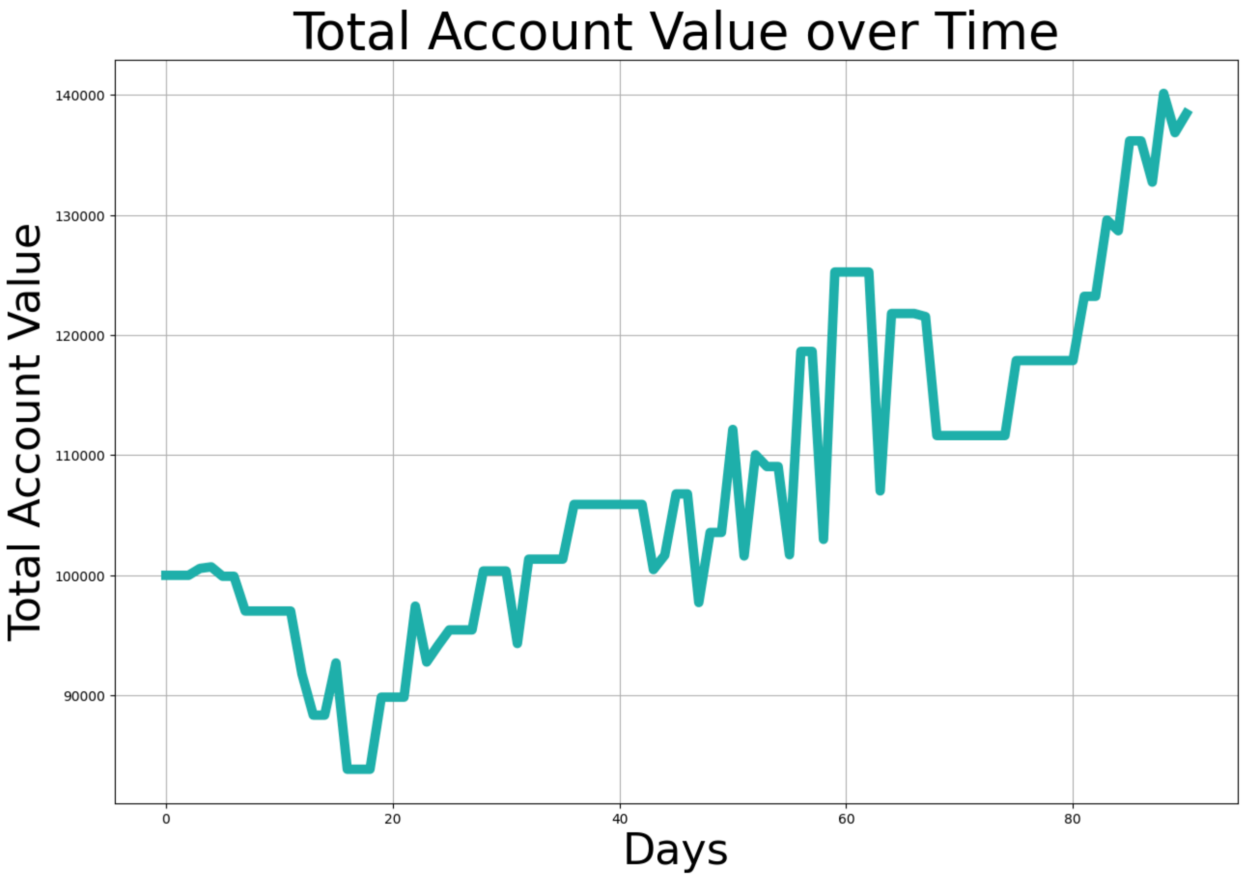
A2C was chosen for the Stock Trading Environment because of its effectiveness in managing environments with continuous states and discrete actions. It handles the unpredictable nature of stock prices well and provides stable updates through the Actor-Critic framework. Its capability to process sequential decisions effectively makes it ideal for the complexities of stock trading.

Results Using A2C in the Stock Trading Environment

The displayed graph of "Total Account Value over Time" illustrates the performance of the A2C-trained agent, showing significant fluctuations and an overall upward trend in account value. These variations highlight moments of substantial gains and some losses, indicating that the A2C has guided the agent to exploit profitable trading opportunities, thus achieving a generally profitable outcome over the period.

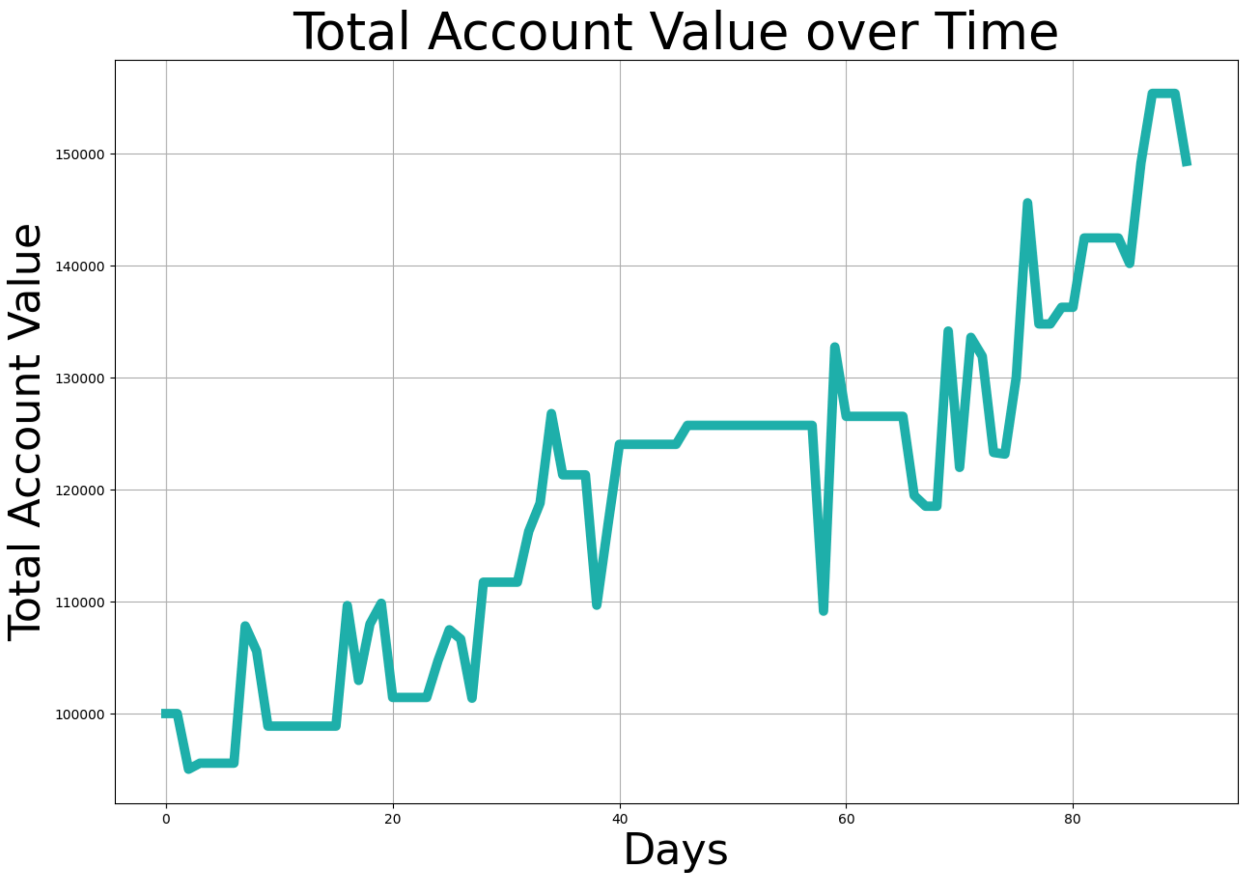

Total Account Value over Time
Total Account Value
150000
140000
130000
120000
110000
100000
90000
0
20
40
60
80
Days

DQN Algorithm



The provided graph does not appear to have any direct connection to or provide information about the Deep Q-Network (DQN) algorithm. The graph displays the Total Account Value over Time, which seems to represent fluctuations in an investment portfolio or account balance over a period of days.

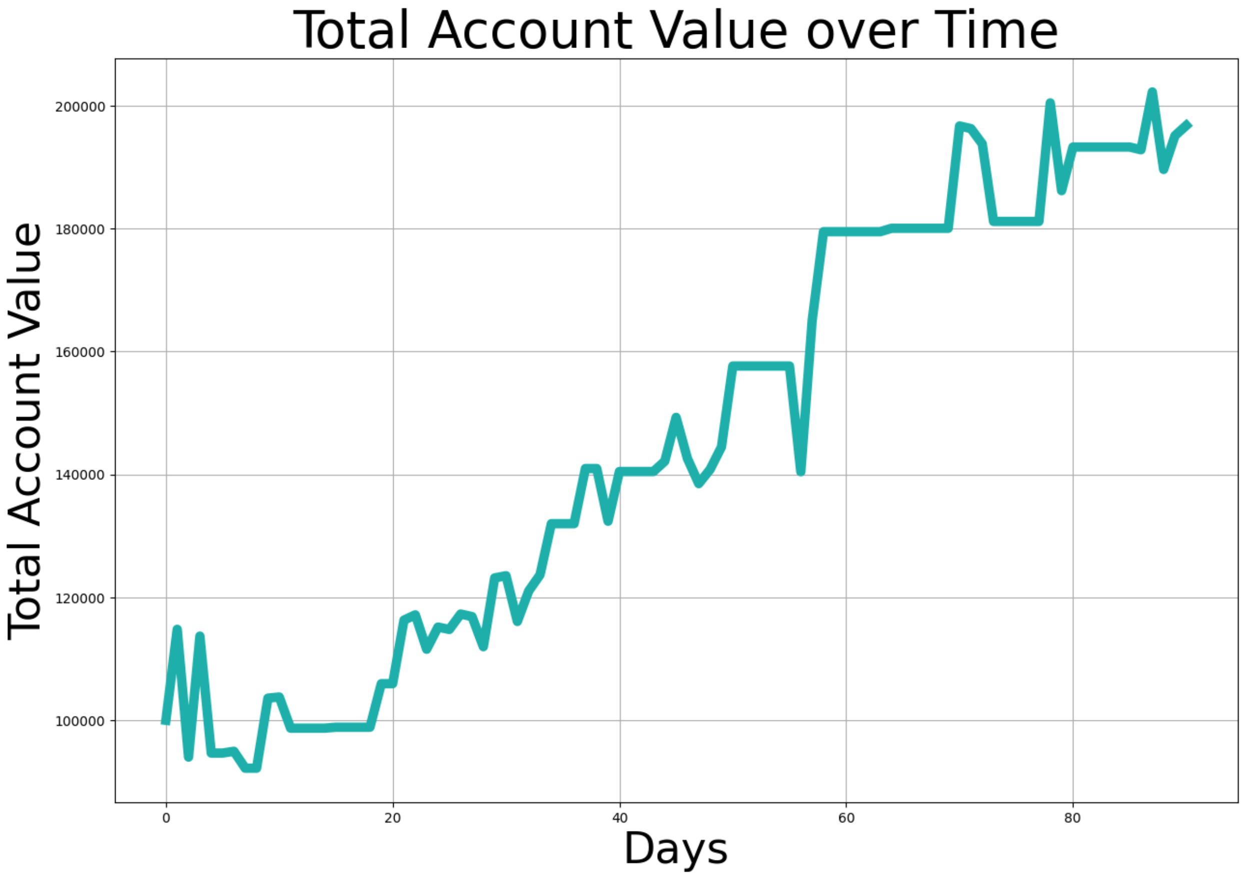
Double DQN



Compared to DQN improved version gives better results which is indicating that over 100 days it almost reached to 150,000,

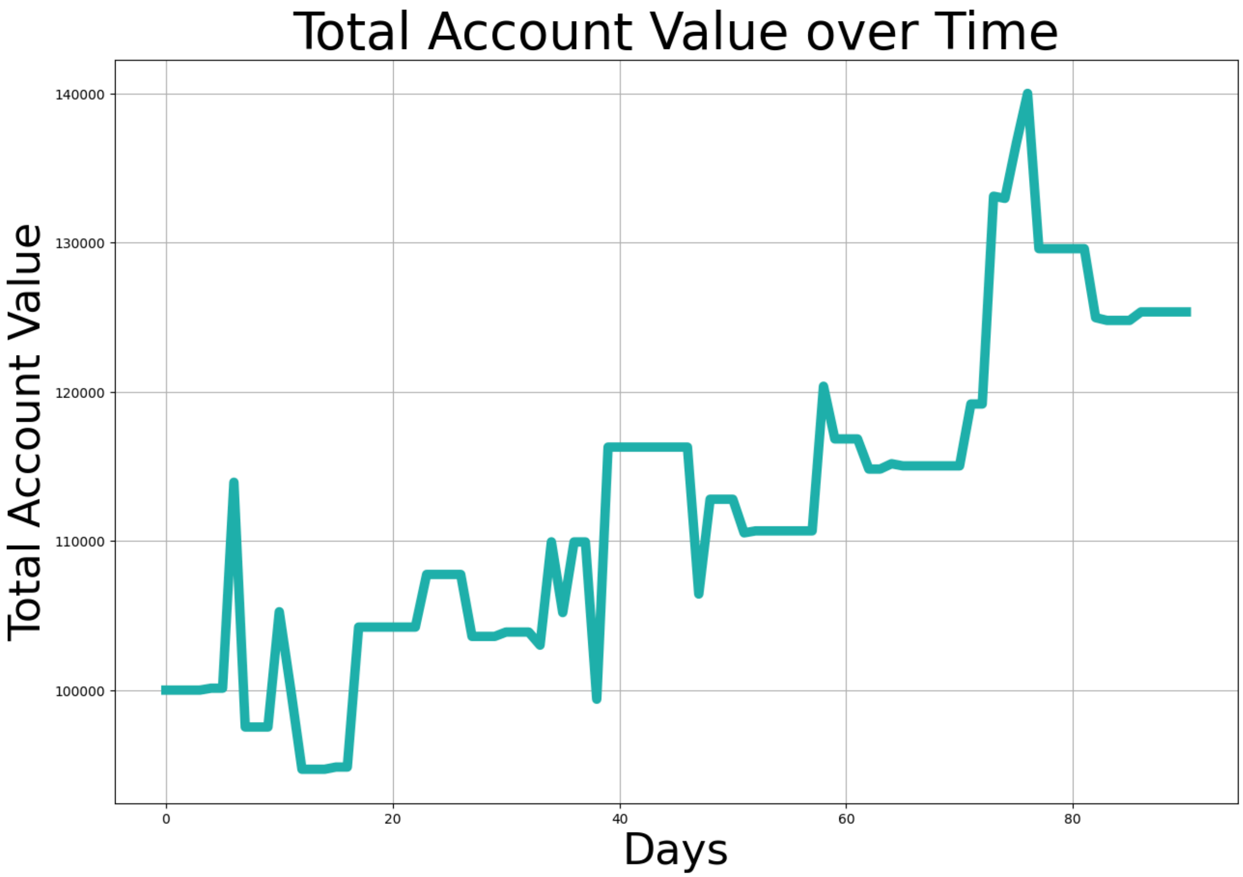
Improved Algorithms:

Proximal policy optimization (PPO)



After the initial fluctuating period, there is a clear trend of growth in the account value. The agent seems to have identified a profitable strategy or set of actions that consistently increase the account value, moving from around $120,000 to over $180,000
It is even touching 2000,000 which performed better compared to all other algorithms.

Prioritized Experience Replay (PER)



This graph depicts the Total Account Value over Time, measured in days. The vertical axis represents the account value, while the horizontal axis shows the number of days.

The graph illustrates significant fluctuations in the account value over time, with several peaks and troughs. The account value experienced sharp increases and decreases, indicating periods of growth and decline.

One notable observation is the presence of several local maxima, where the account value reached its highest points. The most prominent peak occurs around day 70, suggesting a substantial increase in account value during that period.

AWR (Advantage Weighted Regression)

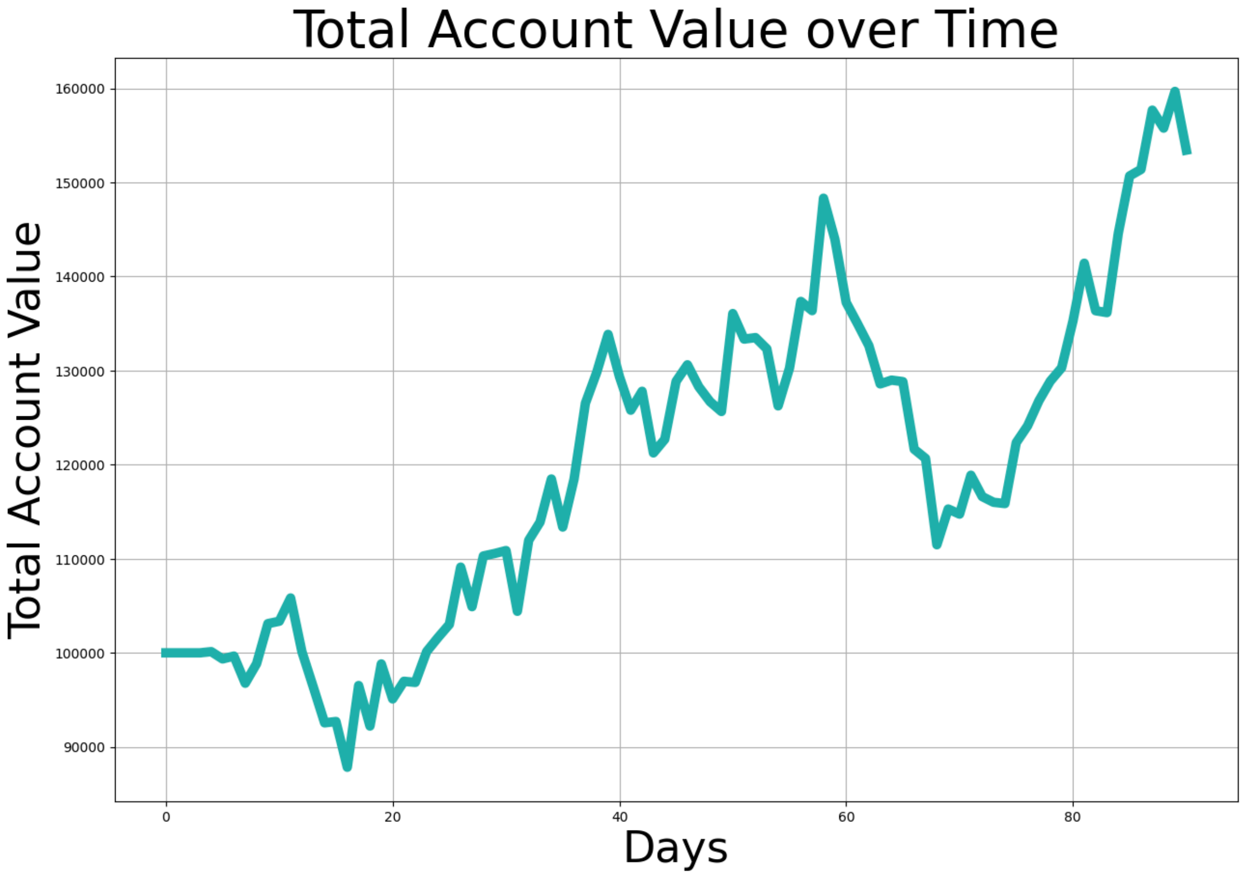
AWR, or Advantage Weighted Regression, is a reinforcement learning strategy that optimizes policies by focusing on the most successful experiences. It selectively learns from outcomes with higher-than-average returns, effectively honing in on the best strategies.

Reason for Choosing AWR for the Stock Trading Environment

AWR was selected for the Stock Trading Environment for its ability to prioritize high-reward states and actions, which is crucial in navigating the volatile stock market. This focus helps maximize performance by capturing exceptional trading opportunities.

Results Using AWR in the Stock Trading Environment

The graph using AWR shows a robust upward trend and effective recoveries from downturns, suggesting a strong strategy. AWR's method of emphasizing advantageous outcomes has proven to be more effective than A2C in this setting, providing a higher profitability and demonstrating better suitability for managing stock market complexities.



TD3 (Twin Delayed Deep Deterministic policy gradient)

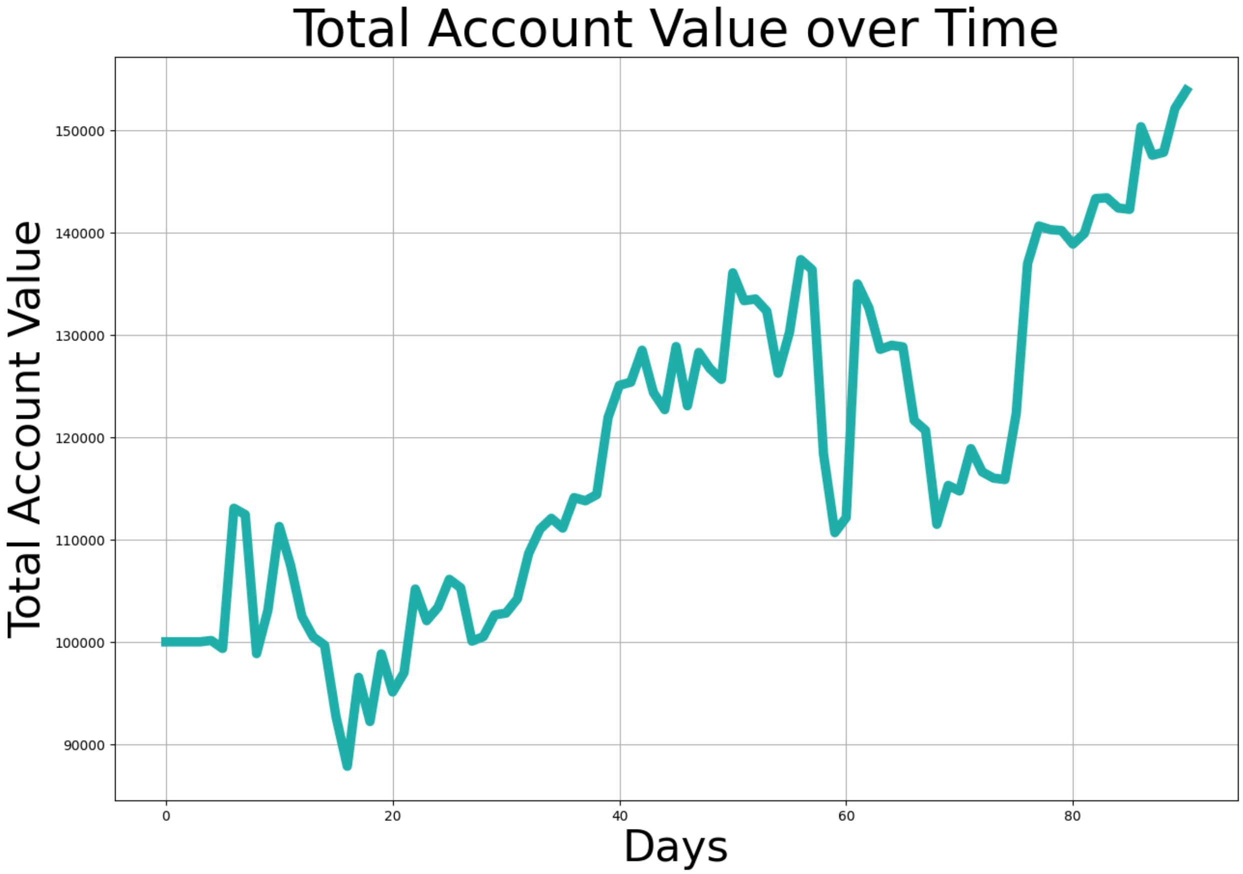
TD3, or Twin Delayed Deep Deterministic policy gradient, is a sophisticated reinforcement learning algorithm that improves on DDPG. It features twin Q-networks to reduce value overestimation, delayed policy updates for learning stability, and smoothed target policies to enhance generalization. These enhancements help develop a more reliable policy that better handles volatile conditions.

Reason for Choosing TD3 for the Stock Trading Environment

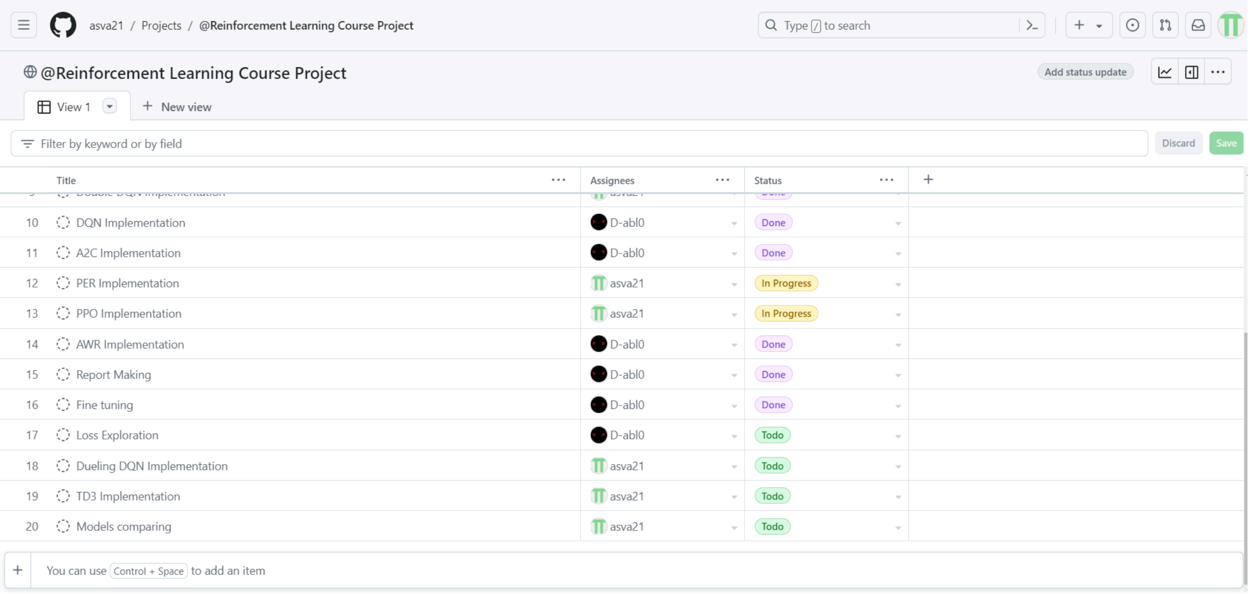
I chose TD3 for the Stock Trading Environment because of its improved stability and precision in estimating action values, which are vital for dealing with unpredictable stock market movements. Its approach to minimizing value overestimation and promoting consistent policy evolution makes it ideal for complex, high-dimensional action spaces typical in stock trading.

Results Using TD3 in the Stock Trading Environment

The chart reflects consistent growth and fewer significant drops, showcasing TD3's effective management of market volatility. The algorithm's advanced features—like delayed updates and policy smoothing—contribute to this performance, demonstrating its superior adaptability and profitability compared to simpler strategies like A2C in this dynamic environment.



Project Management:

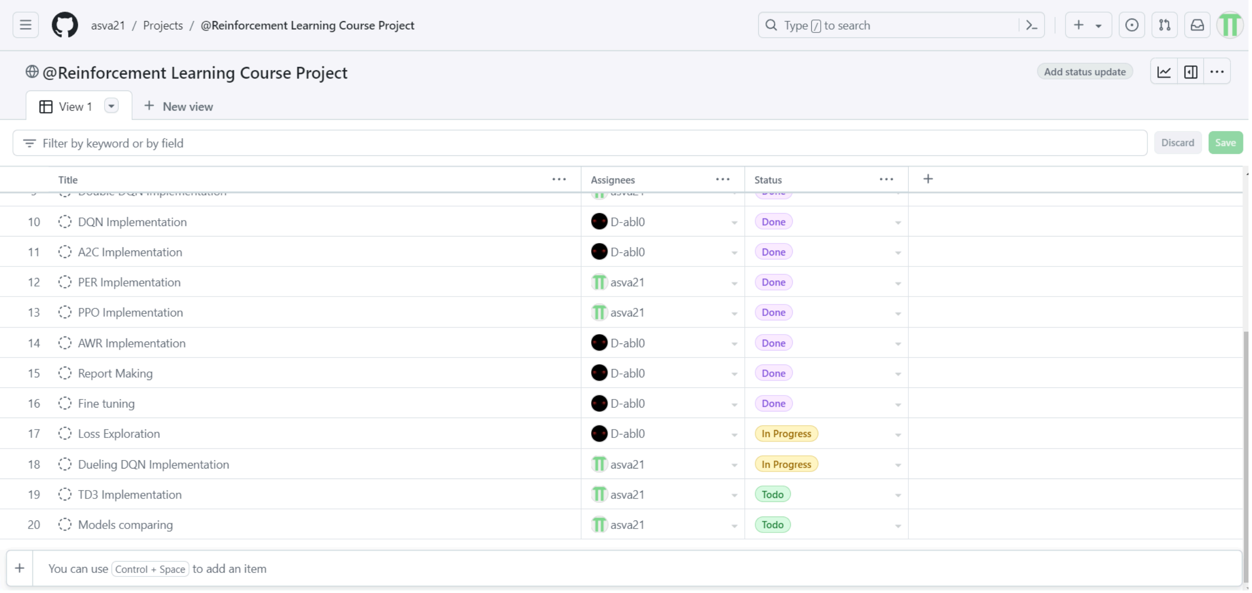


asva21 / Projects / @Reinforcement Learning Course Project

@Reinforcement Learning Course Project

	Title	Assignees	Status
9	Double DQN Implementation	asva21	Done
10	DQN Implementation	D-abl0	Done
11	A2C Implementation	D-abl0	Done
12	PER Implementation	asva21	In Progress
13	PPO Implementation	asva21	In Progress
14	AWR Implementation	D-abl0	Done
15	Report Making	D-abl0	Done
16	Fine tuning	D-abl0	Done
17	Loss Exploration	D-abl0	Todo
18	Dueling DQN Implementation	asva21	Todo
19	TD3 Implementation	asva21	Todo
20	Models comparing	asva21	Todo

You can use Control + Space to add an item

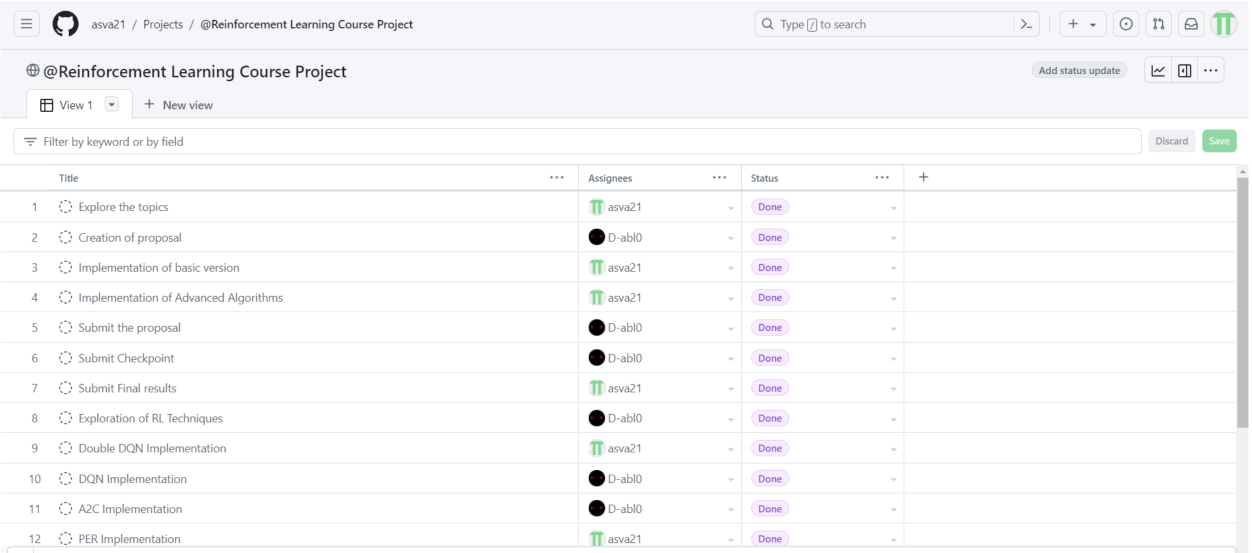


asva21 / Projects / @Reinforcement Learning Course Project

@Reinforcement Learning Course Project

	Title	Assignees	Status
9	Double DQN Implementation	asva21	Done
10	DQN Implementation	D-abl0	Done
11	A2C Implementation	D-abl0	Done
12	PER Implementation	asva21	Done
13	PPO Implementation	asva21	Done
14	AWR Implementation	D-abl0	Done
15	Report Making	D-abl0	Done
16	Fine tuning	D-abl0	Done
17	Loss Exploration	D-abl0	In Progress
18	Dueling DQN Implementation	asva21	In Progress
19	TD3 Implementation	asva21	Todo
20	Models comparing	asva21	Todo

You can use Control + Space to add an item


asva21 / Projects / @Reinforcement Learning Course Project
Type / to search
Add status update
View 1
New view
Filter by keyword or by field
Discard
Save

@Reinforcement Learning Course Project

	Title	Assignees	Status
1	Explore the topics	asva21	Done
2	Creation of proposal	D-abl0	Done
3	Implementation of basic version	asva21	Done
4	Implementation of Advanced Algorithms	asva21	Done
5	Submit the proposal	D-abl0	Done
6	Submit Checkpoint	D-abl0	Done
7	Submit Final results	asva21	Done
8	Exploration of RL Techniques	D-abl0	Done
9	Double DQN Implementation	asva21	Done
10	DQN Implementation	D-abl0	Done
11	A2C Implementation	D-abl0	Done
12	PER Implementation	asva21	Done

Team Member	Project Part	Contribution
sarepall	Code Implementation	50%
cgurram	Code Implementation	50%

Real world application

By integrating RL into the stock trading process, financial institutions and individual traders can potentially enhance their decision-making processes, reduce human errors, and increase profitability through systematic and data-driven approaches. Moreover, the ability to simulate numerous trading scenarios using historical data allows traders to refine strategies under various market conditions without financial risk, preparing them better for actual trading environments. This advancement not only pushes the boundaries of traditional trading practices but also opens new avenues for employing artificial intelligence in financial analytics and investment strategy development, promoting a more efficient and resilient financial market.

We utilized real-world data from NVIDIA Corporation (Ticker: NVDA), obtained from a CSV file named "NVDA.csv". This dataset included daily trading information such as opening and closing prices, highs and lows, and trading volumes. The data was carefully preprocessed to handle any anomalies and normalize values to provide a consistent input for training the reinforcement learning models.

Using this real-world dataset, we explored several RL algorithms including A2C, Dueling DQN, and TD3 within the custom-designed Stock Trading Environment. The results, depicted in various "Total Account Value over Time" graphs, demonstrated that TD3 outperformed others in terms of stability and profitability, effectively managing the inherent volatility of the stock market. This approach not only validated the applicability of advanced RL methods in financial markets but also highlighted the potential of these techniques to improve decision-making in stock trading.