A red logo with a globe and text

Description automatically generated

**Symbiosis Institute of Technology**

Report

For

**AIBF**

**Trading Bot using OpenAI Gym AnyTrading**

**AIML B2 2021-2025**

Utsav Kalra (21070126108)

Mohammed Basumeih(21070126105)

**Problem Statement**  
Stock prices are determined by supply and demand in the stock market, or more precisely, by market forces. In this project, we will focus on developing a reinforcement learning-based trading bot that learns to create profitable trades from historical market data. The basic idea behind this project is to achieve the maximum possible value of cumulative rewards, which represent profits, by taking actions such as buying, selling, and holding stocks, considering the different market situations, transaction costs, and risks.  
a. Need Analysis / Statement of Need  
Given the fact that the stock market situation is very unpredictable and volatile, traders commonly attempt to sell high and buy low to have the maximum profit. However, human decisions are clouded by emotional biases, slow speed of reaction, and a lack of ability to process large amounts of data in an appropriate time scale.  
  
An RL algorithm would be able to automatically make decisions by constantly learning from historical data and adjusting its strategy in accordance with changes in the market. It is capable of making decisions at speeds and accuracies that no human can possibly equal. It has optimality concerning long-term gains because of the exploration of a wide range of trading strategies due to the use of RL.  
Specific needs:  
  
Automated Decision Making: To avoid as much human instinct as possible a bot can arrive at uniform emotional decision-making in trading.  
Handling Huge Data: Financial markets generate enormous amounts of data; a trading bot can evaluate and train through gigantic datasets effectively.  
Profit Maximization: Since the bot learns from historical data, it gains a sense of pattern and trend, ultimately aiding profit maximization over time.  
Risk Management: With reinforcement learning, a bot can learn to become risk-averse by making a balance between short-term rewards and long-term gains.  
b. Technical Functionality  
1. States  
The states in the reinforcement learning environment delineate the current conditions in terms of market and portfolio. The state space can include:  
Current Market Prices: Price of stocks, commodities, and cryptocurrencies.  
Technical Indicators: Popular ones among them includes Moving Averages, Relative Strength Index (RSI), Bollinger Bands, etc.  
Portfolio Position: Different asset holdings and available capitals in hand for trading.  
Supplemental Market Information: News sentiment, earnings announcements, or any other macroeconomic information.  
2. Actions  
At each time step, the agent is capable of making one of three following possible actions:  
Buy: Buys a specified quantity of a particular asset; exhausts the available capital and increases the portfolio holding  
Sell: Sells a specified quantity of the particular asset; increases available capital and reduces portfolio holding.  
Hold: Do nothing and allow the portfolio to be the same as is.  
3. Reinforces  
The reinforcing function is of great importance to the trading behavior of the agent. It is calculated as:  
Profit :Reward for profitable trades 2.  
Loss: Negative reward for unprofitable trades  
Trading Costs:Small penalty to avoid overtrading, an equivalent of brokerage fees or slippage.  
Risk-Adjusted Returns: Rewards might be risk-adjusted using performance metrics such as Sharpe ratio.  
4. Algorithm  
The selected reinforcement learning algorithm is Q-Learning, which is a model-free one to update the Q-values. The latter ones represent the expected reward of the Q-function when taking a particular action in the present state. The update rule is given by

Q(st​,at​)←(1−α)⋅Q(st​,at​)+α⋅(rt​+γ⋅a′max​Q(st+1​,a′))

Where:

* Q(st,at)Q(s\_t, a\_t)Q(st​,at​) is the current Q-value for state sts\_tst​ and action ata\_tat​.
* α\alphaα is the learning rate.
* rtr\_trt​ is the immediate reward after taking action ata\_tat​ in state sts\_tst​.
* γ\gammaγ is the discount factor, representing the importance of future rewards.
* st+1s\_{t+1}st+1​ is the next state after action ata\_tat​.
* max⁡a′Q(st+1,a′)\max\_{a'} Q(s\_{t+1}, a')maxa′​Q(st+1​,a′) is the maximum Q-value for the next state

**5. Discount Factor (γ)**

The discount factor

γ) is the weighting of getting rewards that are located in the future compared to those in the immediate future. In trading, a higher discount factor makes the bot have to aim at future earnings since long-term profitable trades are worth more than short-term ones.

**c. Architecture**

High-Level Overview

The architecture has several components that work together for efficient and profitable trading. Below is the flow of the system:

Market Data and Indicators: Real-time or historical data that the bot uses to observe conditions in the market.

Agent (Trading Bot): Based on the current state-current market prices and indicators-it responds through actions, determining what it should buy, sell, or hold based on its learned policy.

Environment: It simulates the market, its state changing after each action and providing rewards according to the movement in the market.

Algorithm: Q-Learning, an algorithm used to update Q-values by observed reward and transitions with time to change its policy.

Execution: Actions are executed within the system to buy, sell, or hold. The bot learns over time iteratively.

Detailed Architecture Flow Diagram

System Components

Data Collection Layer : Historical market data collection and technical indicator computations

Learning Agent: Core bot, which actually executes the decisions as an action and updates its policy based on a Q-learning algorithm.

Trading Environment: The stock price changes are simulated based on the decisions of the agents and the market conditions.

Action Executor: Buy /Sell action along with the portfolio performance consequence.

Reward Evaluator: The rewards evaluation on the profit/loss along with transaction costs after every trade.

d. Usage / Scope

Usage

Automated Trading: This bot can be put to work for real-time automated stock trading or as a backtesting tool for testing varied strategies in historical data.

Portfolio Management: It can manage multiple assets and will balance buying and selling decisions while having regard for the market conditions and optimize the performance of a portfolio.

Algorithmic Trading: It is applied by the quantitative traders to trade more effectively as well as efficiently with data-driven decision-making.

Scope

Scalability: The model can be scaled across a number of financial markets, such as commodities, forex, or even cryptocurrencies from just stock trading.

Customizable Strategies: Diversified reward functions and technical indicators can be incorporated into the algorithm to customize the bot for trade strategies.

Risk Management: The bot could employ advanced algorithms like Value at Risk (VaR) or a portfolio optimization strategy.

Real-Time Trading: If a brokerage API is integrated into the bot, it would execute real-time trades according to its learned policy.

e. Impact Overview Statement

The proposed trading bot may bring revolutionary change to the traders' relationship with financial markets. With the use of data-driven reinforcement learning, the bot could make better decisions for trades and improve the strategy for trades.

Key effects include:

Increased Profitability: The bot keeps adapting itself to ever-changing market conditions through continuous learning, thus maximizing efficient trading and increasing long-term profits.

Less Human Bias: By automating trades, emotional decision-making is eliminated. Thus, the bot strictly adheres to best strategies without giving in to fear or greed.

Efficiency: It crunches a huge amount of market data and trades faster than any human will, capturing profit-generating opportunities in real-time.

Customizable Strategies: The action space and the reward functions of it are flexible and can be customized based on the risk tolerance and market conditions. This way, it is more versatile for different types of traders.

This way, the bot is trained in risk-averse strategies and makes trades to maximize long-term gains over short-term wins, hence better management of the portfolio.

Results:

A graph with numbers and lines

Description automatically generated A graph with red and green dots

Description automatically generated