National College of Ireland

A Reinforcement Learning Framework to Minimize Reward Latency in Quantitative Trading

MSc Research Project

MS Artificial Intelligence

Sharjeel Nawaz
Student ID: x23389541

School of Computing

National College of Ireland

Supervisor: XXX

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Sharjeel Nawaz		
Student ID:	x23389541		
Programme:	MS Artificial Intelligence	**Year:**	2025 .....
Module:	MSc (Research) Practicum		
Supervisor:	.....		
Submission Due Date:	11-08-2025		
Project Title:	A Reinforcement Learning Framework to Minimize Reward Latency in Quantitative Trading		
Word Count:	19 ..... **Page Count**.....		

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Sharjeel Nawaz .....
Date:	11-08-2025 .....

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

A Reinforcement Learning Framework to Minimize Reward Latency in Quantitative Trading

Sharjeel Nawaz
x23389541

Abstract

Quantitative Trading uses mathematical models to execute trades based on pattern learn from historical stock data. In recent research, most of the mathematical models are reinforcement learning models .However, In reinforcement learning (RL), a key challenge arises from delayed rewards, particularly in quantitative trading environments. For example, the Model may buy a stock but never sell it, so there's no actual reward to evaluate whether the decision was good or not. Therefore, it is difficult for a model to optimize long-term profit by learning from delayed rewards. This presents a significant hurdle in training RL agents effectively for financial market. This research proposes a Reinforcement Learning Framework that uses reward shaping technique to reduce the delay in rewards so the agent can get faster and clearer feedback on the stock purchased.In this research, different deep learning architectures including Deep Q-Networks (DQN) ,Double DQN and Dueling Double DQN was used with our custom reinforcement learning Framework.the purpose of this reward shaping is to discourage excessive inactivity. The introduction of our reward shaping based Reinforcement Learning Framework markedly increased agent activity—by as much as a factor of four relative to unpenalized baselines—and contributed to faster convergence. For the DQN architecture, a penalty magnitude of 5 produced the highest observed profit, reaching 1,400 after 10,000 episodes compared to 350 in the baseline configuration. In the Double DQN architecture, a similar penalty setting yielded a final profit of 1,200, substantially exceeding the 250 achieved without penalization. The Dueling Double DQN architecture attained its highest profit without penalties (from 200 to 1,100 over training), yet still demonstrated measurable gains under appropriately tuned penalty parameters. Overall, these findings suggest that judiciously calibrated penalties can effectively mitigate reward latency, increase decision-making frequency, and improve profitability in reinforcement learning-based quantitative trading systems. However, an excessively high penalty resulted in degrading the model's performance.

This research benefits both academia and industry. For academic research, it contributes a systematic evaluation of reward shaping techniques in RL trading systems. This research provide this systematic evaluation by giving insights into how delayed rewards can be mitigated through carefully tuned penalties. For the financial industry, the research provide a practical method to improve the efficiency and profitability of reinforcement learning based trading agent by encouraging model to make more responsible trading actions according to market changes.

1 Introduction

Reinforcement Learning (RL) has gained significant traction in the field of quantitative trading. The reason for this growing popularity comes from Reinforcement Learning 's ability to make rapid and intelligent decisions such as to (buy or sell) a stock in dynamic Quantitative trading environments. However, a major challenge in applying RL to quantitative trading is the issue of delayed rewards. In simple words, the AI agent makes decisions like buying or selling, but the reward of this action comes later. For example, if it buys a stock, it won't know if the action was good or bad until it sells. This delay makes it hard for the agent to learn, because it cannot clearly understand which actions were good and which were not. This delayed reward not only slows the learning process but also causes instability in training. Moreover, during initial stage model could learn wrong strategy, which can lead to heavy loss. As this issue deals with financial trading, addressing this problem is significantly important.

The aim of this research is to evaluate does our reward shaping based Reinforcement Learning Framework improve the convergence, efficiency and effectiveness of DQN, Double DQN, and Dueling Double DQN agents in quantitative trading?

To address the research question, the following specific sets of research objectives were derived:

1. Review existing reinforcement learning approaches to handle delayed rewards in quantitative trading.
2. Design a custom reward shaping based framework that encourages the model to make trading decisions more frequently and receive rewards.
3. Evaluate the impact of the reward shaping based framework on the action frequency of the reinforcement learning model.
4. Analyse the effect of reward shaping based framework on the profit of the model.
5. Compare the models both on baseline framework and reward shaping based framework.

The major contribution of this research is a Reward Shaping based reinforcement learning framework on which DQN, Double DQN, and Dueling Double DQN has been trained and Evaluated. This frame work decrease reward latency by encouraging the agent to take actions more frequently. Hence, this enables model learn trading strategies more quickly.

Solving this problem is crucial for enhancing the effectiveness and efficiency of Reinforcement Learning models in quantitative trading. Because, it makes trading systems data-efficient, and adaptive to volatile market data quickly. Consequently, this will increase model trading performance. This problem is focus of recent research, and studies have attempted such as reward shaping, and auxiliary tasks to overcome these limitations. However, no prior study has systematically designed and evaluated a reward-shaping framework for DQN-based agents in quantitative trading, particularly examining the effects of varying penalty values on model performance.

The remainder of this report is organized as follows. Section 2 reviews relevant literature on reinforcement learning in quantitative trading, with emphasis on approaches addressing delayed and sparse rewards. Section 3 outlines the research methodology, including the design of the custom reward-shaping mechanism and the selection of model architectures. Section 4 details the design specifications, action space, and reward dynamics, while Section 5 describes the implementation environment, training configurations, and hyperparameter tuning process. Section 6 presents the evaluation plan, experimental results, and discussion. Finally, Section 7 concludes the study and outlines directions for future research.

2 Related Work

Early work on reinforcement learning in quantitative trading primarily focuses on low-frequency environments where market data is aggregated daily or monthly. Moody et al. foundational [1] work implemented Recurrent Reinforcement Learning (RRL) to train trading systems on monthly S&P 500 and T-bill data, directly optimizing financial parameters like the Sharpe ratio and their proposed differential Sharpe ratio. Their technique illustrated superior performance over traditional supervised learning approaches, particularly under state-dependent transaction costs, affirming the strength of RL in sequential, path-dependent decision settings. Conversely, Sarkar [2] employed a Deep Q-Network (DQN) model, also in a low-frequency setting, using technical indicators as inputs for trading a single stock. While both studies showcase the viability of RL for financial decision-making, they differ in modeling scope and depth: Moody implements a portfolio allocation framework with macroeconomic features over a 45-year dataset, while Sarkar adopts a narrower focus on single-stock prediction with limited temporal generalization. Critically, both approaches optimize dense reward structures, relying on cumulative returns or Sharpe-based rewards available at the end of an episode. As such, they sidestep the core challenge of delayed or sparse reward attribution, which remains a barrier to deploying RL systems in more realistic, real-time trading environments. This highlights the need for networks like the one proposed in this research that address reward signal delay directly and enables faster policy convergence under dynamic market settings.

After the advent of DRL, researchers started to explore their uses in financial markets. For instance, Jin et al. [3] introduced a Mean-VaR based actor-critic approach that employs quantile regression to learn the full distribution of returns, efficiently incorporating tail risk to solve the delayed reward problem. Additionally, Huang et al. [4] developed a self-rewarding

framework that dynamically selects the highest of expert-defined or predicted rewards, allowing more adaptive and reward-aligned learning. While both approaches improve robustness under uncertain market circumstances, the former architecture assumes stable return distributions, limiting adaptability in volatile situations. Similarly, the later technique, though flexible, is constrained by its single-asset scope and reliance on expert reward structures, which may reduce generalizability across broad market contexts.

Ensemble learning has also shown great promise in improving the robustness and profitability of RL-based trading systems. In this context, Carta et al. [5] present a multi-layer, multi-ensemble technique that merges deep learning and deep RL algorithms through stacked CNNs, a RL meta-learner, and a final majority voting ensemble. This network emphasizes diversity across model types and learning layers to generate stable trading signals. In contrast, Leem and Kim [6] introduce an action-specialized expert ensemble where separate expert models are trained for buy, hold, and sell actions, each with tailored reward functions. These are merged using soft voting and extended discrete action spaces to enable finer control over trade quantities. While Carta et al. focus on architectural ensembling for robust signal fusion, Leem and Kim exploit functional specialization to model nuanced trading behaviors. Both methods outperform traditional baselines, highlighting different strategies to harness ensemble learning for improved decision-making in noisy financial markets.

A comparative analysis of the frameworks proposed by W. Zhang et al. [7] and Takara et al. [8] reveals distinct approaches to addressing dynamic market conditions and reward sparsity in quantitative trading through RL. For instance, W. Zhang et al. designed a high-frequency trading network under China's T+1 constraints, employing inverse RL to dynamically optimize trading strategy parameters in real time. Their techniques sort out market non-stationarity by rolling training of prediction models and using reward-enhanced Upper Confidence Bound (UCB) models to adapt parameter choices. While this system exhibits strong adaptivity and deployment feasibility, it does not explicitly tackle the sparse or delayed nature of rewards, as rewards are continuously shaped through inverse models and bandit feedback. On the other hand, Takara et al.'s Extended Trading Deep Q-Network (ETDQN) explicitly deals with the sparse-reward issue by recreating reward signals to occur only at trade liquidation, thus mimicking the real-world nature of delayed reward. By integrating distributional learning and sub-goal prioritization, ETDQN lowers the dependency on dense reward signals and shows good performance in volatile situations. Although both networks aim to optimize decision-making in complicated trading settings, Zhang's work aligns more directly with the present research objective by confronting the temporal misalignment between actions and rewards, a critical barrier in training RL agents for financial tasks. Integrating the delayed feedback structure of ETDQN with adaptive mechanisms like those in Takara's approach could represent a promising direction for developing more sample-efficient and risk-aware trading agents.

Recent breakthroughs in multi-agent reinforcement learning (MARL) have presented new methods for strategic quantitative trading, yet their treatment of reward feedback varies significantly. Zhang et al. [9] proposed two MARL techniques constant proportion portfolio insurance multi-agent deep deterministic policy gradient (CPPI-MADDPG) and time-invariant portfolio protection (TIPP-MADDPG) that merge classical financial heuristics into the

MADDPG network. By encoding capital preservation strategies into agents' action constraints and optimizing long-term reward signals, these networks improve risk management and adaptability. However, their reward formulation is still tied to end-of-episode cumulative returns, limiting real-time feedback granularity and leaving the core issue of delayed credit assignment unresolved. In contrast, Shavandi et al. [10] presented a novel hierarchical MARL technique where each agent trades on a different timeframe and knowledge flows top-down across temporal layers. This design not only stabilizes learning across noisy, multi-scale signals but also introduces a reward freezing mechanism where agents hold actions for a fixed duration to simulate realistic delayed feedback. Although both studies underscore the utility of MARL in navigating market uncertainty, Shavandi's approach more explicitly models reward delay as part of the agent's decision loop. This aligns more closely with the present research objective of improving sample efficiency and credit assignment in sparse-reward financial environments. Thus, combining Shavandi's temporal abstraction with Zhang's risk-aware policy design may offer a novel direction for building RL frameworks that better handle real-world reward sparsity.

Tsantekidis et al. [11] have made significant contributions to addressing the challenges of training stable and profitable deep reinforcement learning (DRL) agents in financial markets, proposing two distinct approaches grounded in reward design and policy optimization. In their study, they tackle the instability arising from noisy and delayed profit-and-loss (PnL) rewards by introducing a price trailing-based reward shaping technique. This method conceptualizes trading as a trajectory-following task, where the agent is rewarded for aligning its price trajectory with that of the market, thereby smoothing the reward signal and improving learning stability. The approach is further supported by market-wide training across multiple currency pairs and the use of LSTM-based architectures to capture temporal dependencies, evaluated using both Double DQN and PPO frameworks. In contrast, Tsantekidis, Passalis and Tefas [12] shifts focus to enhancing policy learning through a novel distillation technique. Here, student agents are trained to imitate an ensemble of teacher agents, each optimized on distinct subsets of currency data. This diversity-driven distillation not only stabilizes training in stochastic settings but also improves the generalizability of the resulting trading policies. While the study rewards sparsity and variance through engineered reward functions, the second mitigates optimization noise by leveraging structured knowledge transfer. Together, these works present complementary solutions reward shaping and policy distillation that enhance the robustness and effectiveness of DRL agents in complex financial trading environments.

Zhou et al. [13] proposed the reward-driven Double Dueling Q-learning (R-DDQN) where they combine reinforcement learning and human feedback to improve trading strategies. A lot of traditional reward systems rely on domain expertise, often to the detriment of their performance. This is addressed by R-DDQN, which uses a reward-function network that is trained not only on expert demonstrations but also optimized through a classification-driven process. When evaluated on HSI, IXIC, SP500, GOOGL, MSFT and INTC in the dataset R-DDQN outperformed all other techniques with a return rate of 1502% after 24 months. This is a great progress in trading strategy optimization. Giorgio [14] investigated applying Double DQN (DDQN) and Dueling Network Architecture for financial trading utilizing SP500 history.

The research was looking for agents that could find trading policies robust to transaction costs. Extended commissions significantly changed the pattern of rewards. Despite the rather limited amount of data used, the agents found strategies with overall a good Sharpe ratio and they all performed better than random trading, promising great potential to reinforcement learning algorithms when applied to financial modeling while reminding humans that its hard to craft policies.

Tabaro et al. [15] tries to answer this question by diving into how Deep Reinforcement Learning (DRL) can improve algorithmic trading, Next, they used a Double Deep Q-Network (DDQN) agent to learn the optimal trading policies and directly utilized text analysis of financial statements in the Markov Decision Process (MDP). The results were positive — a 70% improvement in total cumulative reward and increasing the Calmar ratio from 0.9 to 1.3. Our DDQN agent was consistently able to outperform a buy and hold baseline, suggesting that DRL may indeed be the key towards drastically enhancing trading systems. Gao [16] pitted against the Deep Q-Networks vs Double Deep Q-Networks (DDQN) for forecasting stock market. The models both used historical data to predict future prices which would impact how the fund was trading. Even though both versions performed well, the DDQN had an advantage because target networks stabilize training. The research is particularly interesting for its implications on financial forecasting and the growing role of reinforcement learning in finance.

Shi et al. used [17] a deep reinforcement learning model, trying to predict price changes and perform optimal stock market transactions. Their model is based on traditional supervised learning methods with a custom reward function and policy network to learn complex patterns in stock data, which can fail when it comes to long-horizon forecasting. Their results showed DRL performed better than traditional methods, highlighting even more the use of how DRL can forecast stock-market trends. Zhao et al. brought TLRS (Trajectory-level Reward Shaping) [18], an attractiveness to bring more informative rewards into the reinforcement learning domain. Second, TLRS solves the problem of sparse rewards, which can complicate exploration and destabilize learning by encouraging or discourage trajectories that encounter sparse eventsIn the same way as BC, actions are computed with respect to a priori expert-designed expressions and may be positively reinforced if they resemble them. Results in experiments on Chinese and U.S. stock indices demonstrated that TLRS led to a significant enhancement of the predictive performance, with an increase of 9.29% for Rank Information Coefficient, and also improved the computational efficiency.

Zia Uddin, 2019 [19] - evaluated the vulnerabilities in RL models that adopt reward shaping particularly under adversarial attacks. To be specific, malicious actors can interfere with reward feedback, introducing noise that affects what the entity ultimately learns. The author suggested adopting the K-Nearest Neighbors (KNN) algorithm as a defense mechanism to protect the reward signal from corrupted data. Although noise may lower performance, the KNN-based defense allowed to reduce this effect: as a result of adding noise, the agent appeared to be more robust. Yuan et al. Another approach towards improving exploration was introduced by [20] who proposed providing adaptive intrinsic rewards using what they called AIRS (Automatic Intrinsic Reward Shaping). AIRS achieves high performance on general RL benchmarks such as MiniGrid and the DeepMind Control Suite, but its application in financial domains is still

very limited. While AIRS selects shaping functions with the help of real-time task-return estimates (helping reduce biases in objective functions), there has been limited use of shaping functions in finance. Rodinos et al. [21] has introduced a reward scheme based on Sharpe ratio that is tailored to DRL agents for financial trading. The profits that DRL agents can rake in are potentially huge — but so is the downside risk of the markets that they need to compete against. These risk factors are completely left out of the conventional reward schemes based on PnL. Tested with Speedlab AG data, the introduction of the most commonly used measurement method for risk-adjusted returns (the Sharpe ratio) further improved metric performance across multiple metrics as provided by authors.

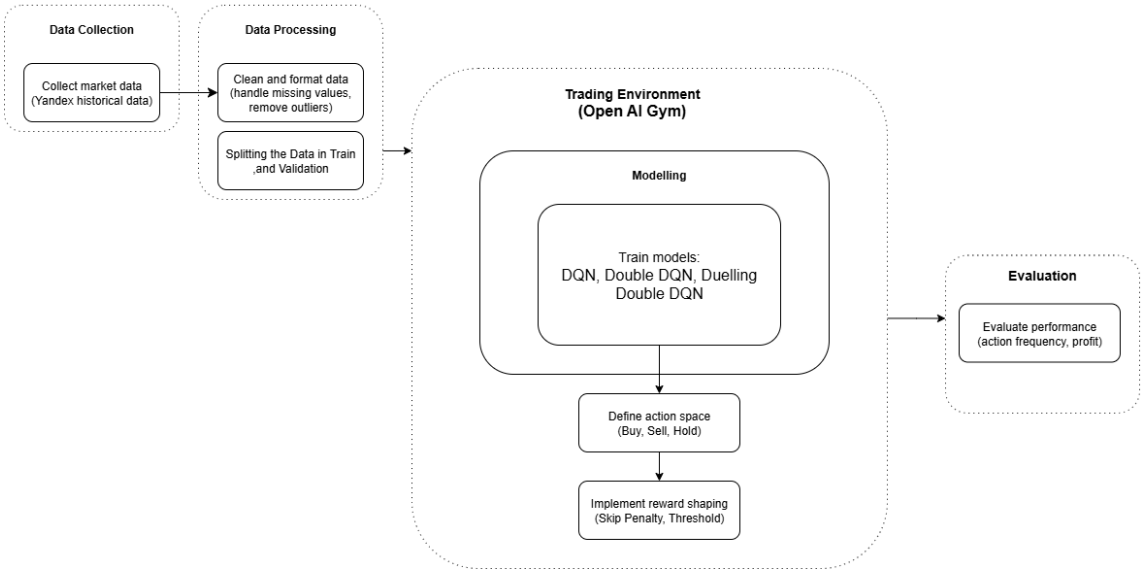
Kölle et al. Later on, a mechanism was introduced by [22] to cooperative among autonomous agents based on stock-market dynamics. These agents are then able to earn portions of the rewards acquired from other agents, thus aligning individual decisions with a common goal. Experimental results in general-sum Markov games showed that this scheme was conducive for cooperation and worked especially well on complex social dilemmas. Agents dynamically created roles and subdivided tasks as the task complexity increased, indicating that the approach could potentially foster collaboration in multi-agent systems.

All **previous solutions** did not fully focus on delayed reward issue but they try to solve it with methods that are overly complex. Additionally, they fail to assess the impact of penalty magnitude in reward shaping, model learning ability, model efficiency,and model performance.

This research fill all these gaps. It provides an in-depth explanation of how adding inactivity penalties affects the performance and efficiency of DQN, Double DQN, and Dueling Double DQN models. By doing so, this research offers insights for applied researcher how want to design efficient and reward shaping-based trading systems. Moreover, it also explore the relation of the magnitude of penalty value with model performance. Additionally, it also advances academic understanding of how reward shaping can effectively resolve the issue of delayed reward in RL for finance.

3 Research Methodology

Figure below shows the methodology diagram of the our proposed solution.It shows reward shaping based reinforcement learning system by integrating penalty and threshold parameters into the reward function. It will discourage inactivity and improve learning speed in sparse and delayed reward environments.



Figure(1)

3.1 Data Collection

Data set was collected from https://www.investing.com/equities/yandex-historical-data . This was YANDEX Stock data with feature of Date, Time ,Close (Price), Open, High, Low, and Volume.

3.2 Data Processing

In the section pre-processed the data by changing column data types to numeric, filling missing values ,and assigning proper columns name.additioallt ,splitted the data in train and validation.

3.3 Trading Environment

In this phase, a trading environment is simulated using OpenAI Gym.this environment provide real-time trading environment where agent can take action ,and each action give reward feedback and state transitions. In this trading environment, the model can choose between three actions, similar to real-world trading: **Buy**, **Sell**, or **Hold**.

Moreover, around these actions, i implemented custom Reward Shaping based Reward Function.

Reward Function

$$\mathcal{R}_t = \begin{cases} R_t + \lambda, & \text{if } s_t > \tau \\ R_t, & \text{otherwise} \end{cases}$$

Figure 2: Mathematical modeling of the proposed solution.

R(t): Immediate reward at time step t (e.g., reward from the current action).
s(t): Number of consecutive time steps the agent has chosen the **Hold** action.
T - tau: Threshold for maximum allowed consecutive Hold actions (e.g., 3).
λ - lambda: Skip penalty applied when s(t) >T.

A normal reward function in reinforcement learning simply gives the agent feedback based on the direct outcome of its action but this reward function keep track of Agents Actions.if agent stays inactive (holds) for too long in a row, a penalty is applied to encourage it to act more often. This helps the agent learn better by rewarding frequent decisions and reducing inactivity. It is implemented through adding penalty to reward by using the reward function mentioned in the Figure(2).

This penalty(Reward Shaping) ,and how it affects the model will be discussed further in detail in the evaluation section.

3.4 Modeling

During the modeling phase, three reinforcement learning models were explored in the context of quantitative trading: **Deep Q-Network (DQN)**, **Double DQN**, and **Dueling Double DQN**. All models were trained within above proposed trading environment using different penalty values (reward shaping).The purpose of different penalty values is to assess its impact on model performance. The dataset used was YANDEX Stock data (80/20 train-validation split).

3.5 Evaluations

The evaluation aimed to evaluate our reward shaping based reinforcement learning framework affect on convergence and model performance. The metrics used for assessment included the total number of actions taken by the agent over the episodes and the net profit achieved on the validation dataset.Action frequency metric is enough to measure efficiency because if model takes more actions it will converge quickly. Additionally, the quality of these actions will be measured by the profit.All models were evaluated under different penalty values to compare their performance.moreover, the model with penalty zero is serve as baseline ,and it will be compared against different reward shaping values.

4 Implementation

This research includes training of 3 different models i.e DQN, Double DQN and dueling double DQN. For the training a separate virtual environment was set up using Python 3.7, PTan and Pytorch were used to train the models. Due to hardware limitations Cuda was not available and

model training was carried out on CPU. For each model a jupyter notebook was created and the same training / validation and test specifications were used.

Training was carried out by tweaking the hyperparameters based on heuristics. Results were improved by training the models one-by-one over an order of days.

5.1 Programming Language and Tools

Table 1: Summary of Python libraries used

Programming Language	Python 3.7
Frameworks / Libraries	PTan, Tensorflow, Open AI Gym, Pytorch, ignite, datetime and itertools
Data processing and visualizations	Numpy, Pandas, Seaborn, Matplotlib

5.2 Training Configuration

Dataset: Yandex Historical Dataset (link)
Train / Test data : 80% training, 20%validation
Data Format: DATE, TIME, OPEN, HIGH, LOW, CLOSE, VOL

Data has 24 (hourly) time-steps for each day. All other columns of the dataset are normalized. Then the complete dataset is split in an 80-20 ratio for training and validation for model training.

5.3 Experimental Design

All the models, including DQN, Double DQN and dueling double DQN will be trained with penalty values of 0,1,3, and 5.
The penalty value of zero will not have any effect on the reward.In other words, the penalty of value zero represents the standard reinforcement learning framework which will be our baseline.

5.3.1 Selected Hyperparameters

Parameter	Values Tried	Select Value
Batch Size	32, 64, 128, 256, 512	64
Learning Rate	1e-4, 1e-3, 5e-4, 1e-6	1e-4

Gamma	0.50, 0.75, 0.80, 0.85, 0.90, 0.95, 0.99	0.95
Replay Size	1000, 5000, 10,000, 20,000 50,000	50,000
Epsilon Decay	Linear, Exponential	Linear
Target Sync	500, 1000, 5000, 10,000 steps	1000 steps

Table 2: Summary of all hyperparameters tried while experimenting and selected

We tried different learning rates, batch sizes and other hyperparameters' values but selected the ones that showed most generalization on validation data.

6. Evaluation

6.1 Evaluation Plan

To assess the effectiveness of our proposed reward shaping based reinforcement learning framework. we trained models both on a baseline setup (penalty zero) and on our proposed reward shaping based reinforcement learning framework. All the experimental results are discussed below.

Research Question: Does our proposed reward shaping based reinforcement learning framework improve the efficiency and effectiveness of quantitative trading systems?

Evaluation Metrics:

Actions Frequency metric: Measures the number of actions taken by the agent over the episodes.this will tell how the reward shaping encouraging model to take more action because if model will take more action then it is possible that I will converge quickly.
Profit metric: Measures the profit of the model on validation data.the purpose of this metric is to not only the model profit but also evaluate the quality of action taken by model with and without penalty.

Baseline models for Comparison:

The model used for comparisons are the ones where no penalty (no reward shaping) is imposed on the model. In the implementation, their penalty is set to zero so models can be trained on the standard reinforcement learning loss function.

6.2 Evaluation of the proposed Framework based on total number of actions taken by the agent

In this section, I will evaluate three architectures: Deep Q-Network (DQN), Double DQN, and Dueling Double DQN, based on the number of actions (action frequency) taken over the episodes. All these models are trained with different penalty values such as 0,1,3, and 5. All of these models will be evaluated and compared to each other. These models will be discusses one by one.

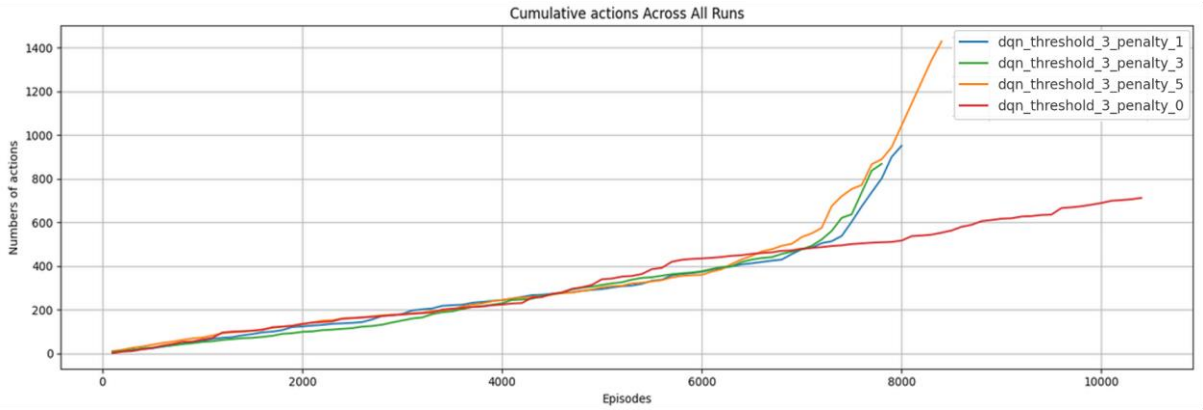


Figure 3: Illustration of actions taken by DQN model with respect to trained episodes

Figure(3) illustrates the cumulative number of actions taken by DQN agents across 10,000 episodes under varying penalty values (0, 1, 3, 5) with a fixed threshold of 3. The unpenalized model (penalty_0) has penalty zero,and considered as baseline. It exhibits the lowest action frequency throughout, indicating a tendency toward inactivity in the absence of reward shaping. In contrast, all penalized variants show a clear increase in cumulative actions, especially beyond episode 7,000. Among these, the model with a penalty of 5 achieves the highest action count, suggesting that moderate penalization effectively motivates the agent to engage more actively with the environment. This trend supports the hypothesis that introducing inactivity penalties promotes more frequent decision-making and exploration, which is critical in overcoming reward sparsity during training.

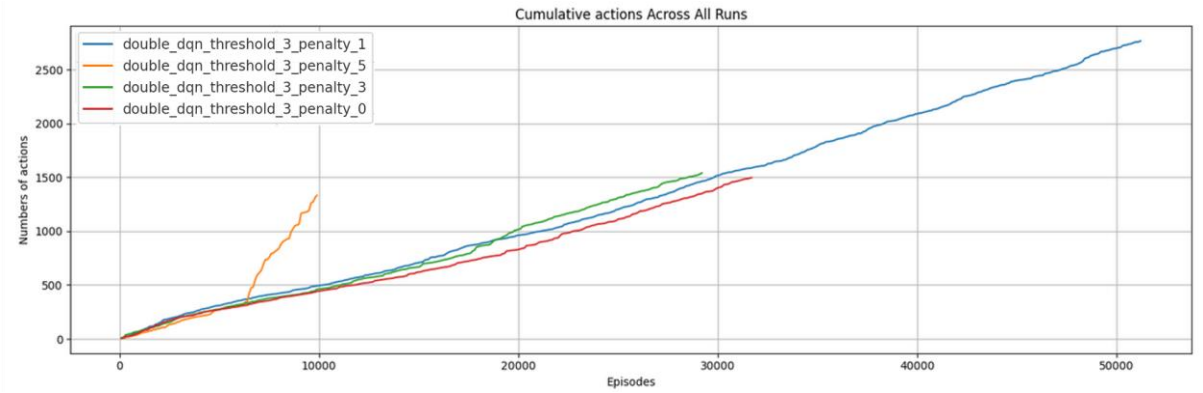


Figure 4: Illustration of actions taken by Double DQN model with respect to trained episodes

Figure(4) shows the cumulative number of actions taken by Double DQN agents over time, evaluated under different penalty values (5, 3, 1, 0) with a constant threshold of 3. The model without penalties (penalty_0) demonstrates steady but lower action growth compared to its penalized counterparts. The penalty 5 configuration results in the highest number of cumulative actions, indicating that moderate reward shaping effectively encourages more active trading behavior. The penalty 1 and penalty 5 models follow similar upward trends, although the run for penalty 5 terminates early, potentially due to instability or convergence issues. These results suggest that, within the Double DQN architecture, appropriately tuned penalties enhance exploration and interaction frequency, but overly aggressive shaping may compromise training stability.

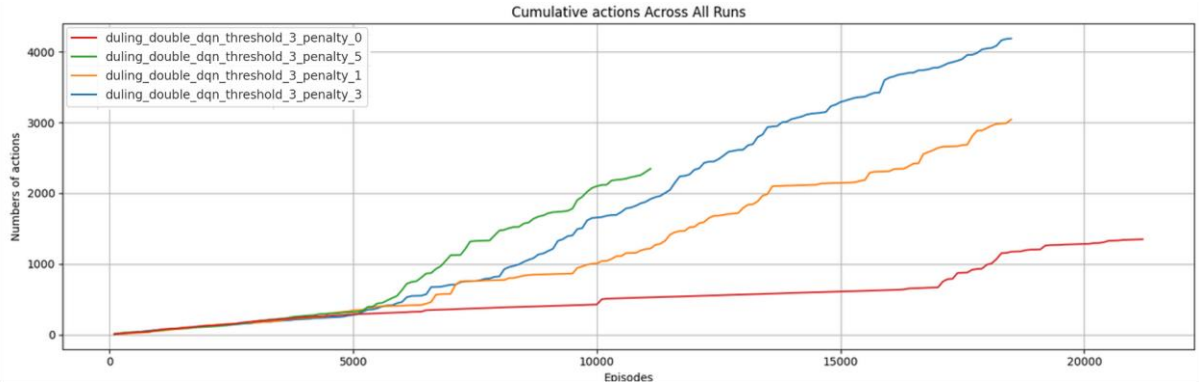


Figure 5: Illustration of actions taken by Dueling Double DQN with respect to trained episodes

The above figure shows the number of actions taken by Dueling Double DQN agents across different penalty settings (5, 3, 1, 0) with a fixed threshold of 3. The model with penalty 3 demonstrates the highest action frequency, followed closely by the 1 and 5 configurations, indicating that introducing moderate to strong penalties significantly boosts agent activity. In contrast, the penalty 0 model (no reward shaping) lags behind with the lowest action count, reflecting a tendency towards inaction when no incentive is provided to discourage holding. These results suggest that the Dueling Double DQN architecture is highly responsive to reward shaping, and that carefully tuned penalties can substantially increase engagement and exploration during training.

Model	Threshold	Penalty	Actions per 5,000 Episodes	Actions per 7,000 Episodes	Actions per 10,000 Episodes
DQN	**3**	**0**	**250**	**500**	**900**
	3	1	260	530	2400
	3	3	270	570	3000

	3	5	320	610	3600
Double DQN	**3**	**0**	**200**	**350**	**420**
	3	1	210	370	470
	3	3	230	410	500
	3	5	220	400	1300
Dueling Double DQN	**3**	**0**	**400**	**440**	**500**
	3	1	410	1500	2100
	3	3	410	750	1700
	3	5	430	760	1000

Table 3: Comparison of number of actions taken by DQN, Double DQN and Dueling Double DQN agents after 5000, 7000 and 10000 episodes of training

The above table represent the same data about the number of action (action frequency) which is represented in above figures 3,4,5.it shows that when the penalty is set to zero the models perform the worst and action frequency is quite low. Whereas when the penalty threshold and reward increase the model's action frequency starts to improve drastically. Even in a simple DQN, the model action frequency starts to increase exponentially after only 5000 episodes whereas the rate of growth of non-reward-spaced still remains stagnant throughout. Similar trends are observed with Double DQN and Dueling Double DQN agents.

Therefore it is concluded that reward shaping encourages the model to make more actions.in other words, it increases model action frequency. This proves that there is a direct relation of magnitude of penalty with the number of action.

6.3 Evaluation of the proposed Framework based on model Profit

In this section, I will evaluate three architectures: Deep Q-Network (DQN), Double DQN, and Dueling Double DQN, based on profit metric. The purpose of this evaluation is to analyse the impact of reward shaping on profit.in other words,we want to analyse how actions quality is impacted by reward shaping because in reward shaping we are encouraging model to take actions. All these models are trained with different penalty values such as 0,1,3, and 5. These models will be discusses one by one.

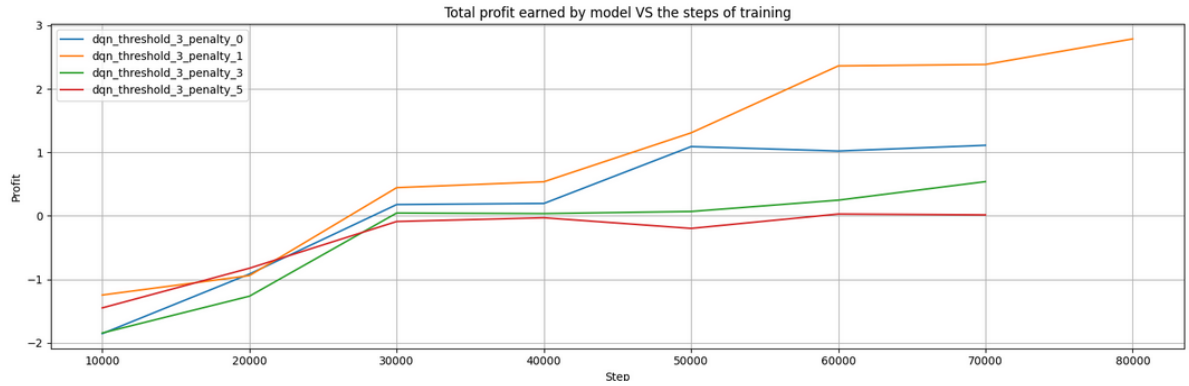


Figure 6: Illustration of profit earned by DQN model with respect to trained episodes

This plot illustrates the cumulative profit trajectories for the vanilla DQN agent under varying penalty settings (0, 1, 3, 5). The model with the penalty of 1 clearly outperforms others, showing a consistent upward trend in cumulative rewards. The model with zero penalty performs second best.The model with the highest penalty suffers significant degradation in performance, showing negative cumulative rewards throughout.

This suggests that over reward shaping discourages exploration and undermines learning in standard DQN setups.

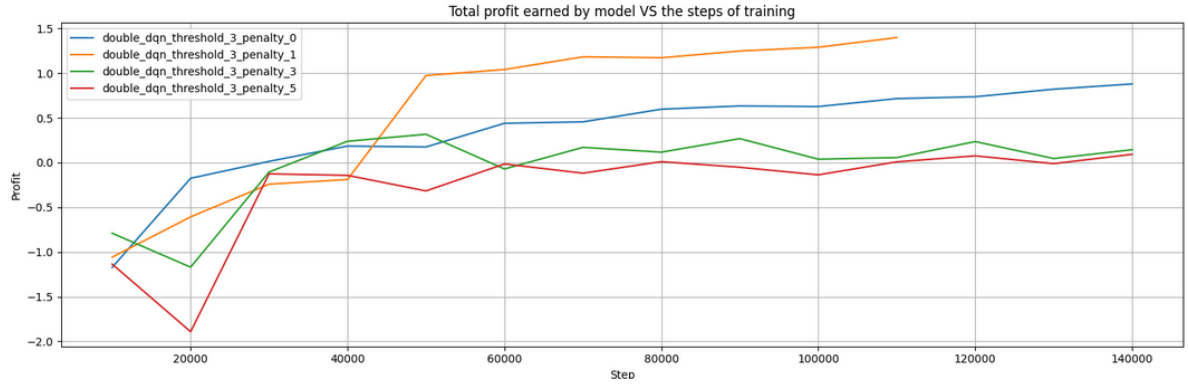


Figure 7: Illustration of profit earned by Double DQN model with respect to trained episodes

The plot shows the performance of the Double DQN agent across the same penalty configurations. Unlike the DQN model, mild penalties appear to slightly improve performance over the baseline (penalty 0), with the 1 penalty yielding the most consistent gains over time. However, extreme penalties (5) again degrade performance, though not as severely as in the DQN case. This supports the idea that Double DQN's improved value estimation helps it better tolerate moderate reward shaping, but remains sensitive to overly aggressive penalization.

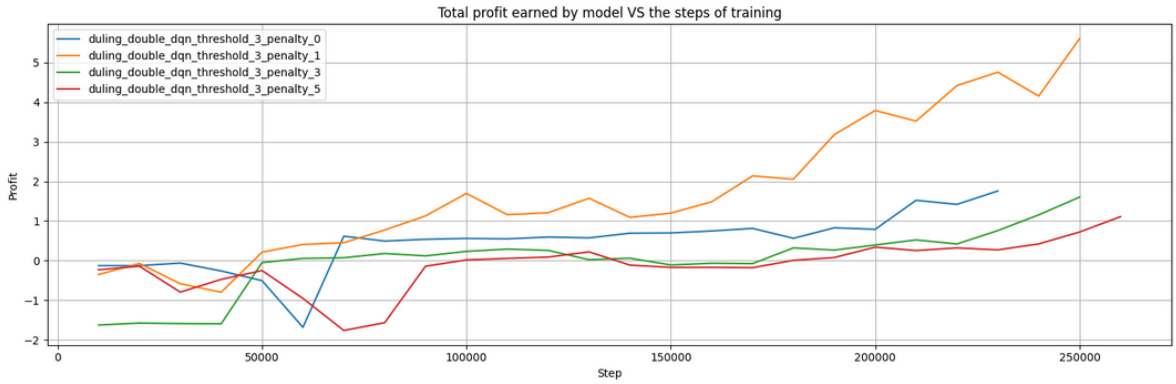


Figure 8: Illustration of profit earned by Dueling Double DQN model with respect to trained episodes

The plot presents cumulative profits for the Dueling Double DQN agent. Similar to other models, the model with penalty 1 achieves the highest performance, with a clear and steady increase in cumulative rewards. The model with penalty 5 once again performs the worst, especially during early training. The model with the penalty of 1 clearly outperforms others.These results suggest that Dueling architectures, which separately estimate state value and advantage. This separate estimations ,may naturally promote more effective action selection which eventually increases model performance.

Model	Threshold	Penalty	Profit @ 5,000 Episodes	Profit @ 7,000 Episodes	Profit @ 10,000 Episodes
DQN	**3**	**0**	**-100**	**150**	**350**
	3	1	200	400	650
	3	3	400	600	700
	3	5	900	1100	1400
Double DQN	**3**	**0**	**-600**	**-100**	**250**
	3	1	-400	-50	400
	3	3	-100	300	500
	3	5	200	400	1200

Dueling Double DQN	**3**	**0**	**-400**	**-200**	**200**
	3	1	-100	100	300
	3	3	-100	200	600
	3	5	-100	600	1100

Table 4: Comparison of profit earned by DQN, Double DQN and Dueling Double DQN agents after 5000, 7000 and 10000 episodes of training

This table represents the same data which is represented in the above figures (6,7,8) In the above table, it can be seen that profit in the initial episodes stays negative as the model is making guesses in the initial stage. And as soon as the model starts to mature the profits start to increase exponentially. however,it can be seen in every model that significant high values of penalty degrade model profit.

From all above experiments, it is evidently clear that increase in the penalty value increase the model profit. This increase is due to more trade actions which result in quick model convergence ,and more trades. However,if we increase the penalty significantly then model profit degrades significantly.

6.4 Discussion

This research proves that rewards shaping helps in providing a conclusive enhancement in action frequency as soon as even a small amount of penalty is associated with the model. specifically, that there is a direct relation between magnitude of penalty and the number of action.in simple words, if we increase the penalty value action frequency will increase. Although the rate of increase varies across the models, the overall trend is clear. This different in trend is due to the different architectural natures of models. DQN was the most sensitive to penalization, with action frequency increasing sharply as the penalty magnitude rose, suggesting a high dependence on external incentives for active trading. In contrast, Double DQN showed more gradual improvements, indicating that its stability-oriented design moderates the behavioural impact of penalties. Dueling Double DQN differed further, achieving strong results even without penalties.

Now, we discuss the relation between penalty and model profit. It can be clearly seen that reward shaping impacts the model's effectiveness. Moderate increase in the penalty value increase the model profit. This increase is due to more trade actions. An increase in actions boosts profit both directly and indirectly, as more actions lead to higher profit and also enable the model to converge more quickly which result in quick convergence. However, if we increase the penalty value significantly high then it decrease the model profit.This trend is observed for all the models because instead of learning the effective trading strategy, it start

doing more and more action due to extreme influence of reward shaping. Although this trend appears in all models, the extent of its impact varies depending on each model's architecture.

From all the experiments I can conclude that reward shaping helps model to converge quickly ,and also increase the model performance. However, it must be applied moderately and with domain knowledge. As we have observed excessive reward shaping can degrade the model's performance.

My results agree with previous studies: Reward Shaping can improve the performance of reinforcement learning models.However, if the reward shaping or penalty is too strong, it can cause the model to take too many actions and fail to learn an effective strategy.

7. Conclusion and Future Work

This study set out to tackle one of the key challenges in applying reinforcement learning (RL) to quantitative trading: the issue delayed rewards. By introducing a reward-shaping based reinforcement Learning Framework that penalizes prolonged inactivity. In experiments we used DQN, Double DQN, and Dueling Double DQN architectures. Fortunately,the results showed that the proposed framework not only improved convergence speed but also led to more profitable.However,extremely penalty value decreased the model performance.which conclude that the intensity of reward shaping plays significant role. The findings highlight the importance of reward-shaping based reinforcement Learning Framework in building more responsive and intelligent trading agents.

In future research, the reward shaping framework can be extended in several ways.First, we can direct our research to adaptive penalty-based reward shaping.This will help in balancing action frequency with effective strategy learning. secondly,we can advance research in the direction where secondary dataset has to be used to determine the optimal reward shaping parameters. we can experiment this framework with other reinforcement learning model and see its impact on them. Finally, training and evaluating this framework on a larger financial dataset.

References

[1] J. E. Moody and M. Saffell, "Reinforcement Learning for Trading".

[2] S. Sarkar, "Quantitative Trading using Deep Q Learning," Int. J. Res. Appl. Sci. Eng. Technol., vol. 11, no. 4, pp. 731–738, Apr. 2023, doi: 10.22214/ijraset.2023.50170.

[3] B. Jin, "A Mean-VaR Based Deep Reinforcement Learning Framework for Practical Algorithmic Trading," IEEE Access, vol. 11, pp. 28920–28933, 2023, doi: 10.1109/access.2023.3259108.

[4] Y. Huang, C. Zhou, L. Zhang, and X. Lu, "A Self-Rewarding Mechanism in Deep Reinforcement Learning for Trading Strategy Optimization," Mathematics, vol. 12, no. 24, p. 4020, Dec. 2024, doi: 10.3390/math12244020.

[5] S. Carta, A. Corriga, A. Ferreira, A. S. Podda, and D. R. Recupero, “A multi-layer and multi-ensemble stock trader using deep learning and deep reinforcement learning,” Appl. Intell., vol. 51, no. 2, pp. 889–905, Feb. 2021, doi: 10.1007/s10489-020-01839-5.

[6] J. Leem and H. Y. Kim, “Action-specialized expert ensemble trading system with extended discrete action space using deep reinforcement learning,” PLOS ONE, vol. 15, no. 7, p. e0236178, Jul. 2020, doi: 10.1371/journal.pone.0236178.

[7] W. Zhang, T. Yin, Y. Zhao, B. Han, and H. Liu, “Reinforcement Learning for Stock Prediction and High-Frequency Trading With T+1 Rules,” IEEE Access, vol. 11, pp. 14115–14127, 2023, doi: 10.1109/access.2022.3197165.

[8] L. D. A. Takara, A. A. P. Santos, V. C. Mariani, and L. D. S. Coelho, “Deep reinforcement learning applied to a sparse-reward trading environment with intraday data,” Expert Syst. Appl., vol. 238, p. 121897, Mar. 2024, doi: 10.1016/j.eswa.2023.121897.

[9] H. Zhang, Z. Shi, Y. Hu, W. Ding, E. E. Kuruoğlu, and X.-P. Zhang, “Optimizing Trading Strategies in Quantitative Markets Using Multi-Agent Reinforcement Learning,” in ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Seoul, Korea, Republic of: IEEE, Apr. 2024, pp. 136–140. doi: 10.1109/icassp48485.2024.10446489.

[10] A. Shavandi and M. Khedmati, “A multi-agent deep reinforcement learning framework for algorithmic trading in financial markets,” Expert Syst. Appl., vol. 208, p. 118124, Dec. 2022, doi: 10.1016/j.eswa.2022.118124.

[11] A. Tsantekidis, N. Passalis, A.-S. Toufa, K. Saitas-Zarkias, S. Chairistanidis, and A. Tefas, “Price Trailing for Financial Trading Using Deep Reinforcement Learning,” IEEE Trans. Neural Netw. Learn. Syst., vol. 32, no. 7, pp. 2837–2846, Jul. 2021, doi: 10.1109/tnnls.2020.2997523.

[12] A. Tsantekidis, N. Passalis, and A. Tefas, “Diversity-driven knowledge distillation for financial trading using Deep Reinforcement Learning,” Neural Netw., vol. 140, pp. 193–202, Aug. 2021, doi: 10.1016/j.neunet.2021.02.026.

[13] Zhou, C., Huang, Y., Cui, K. and Lu, X., 2024. R-DDQN: Optimizing Algorithmic Trading Strategies Using a Reward Network in a Double DQN. Mathematics, 12(11), p.1621.

[14] Giorgio, B., 2025. Dueling Deep Reinforcement Learning for Financial Time Series. arXiv preprint arXiv:2504.11601.

[15] Tabaro, L., Kinani, J.M.V., Rosales-Silva, A.J., Salgado-Ramírez, J.C., Mújica-Vargas, D., Escamilla-Ambrosio, P.J. and Ramos-Díaz, E., 2024. Algorithmic trading using double deep q-networks and sentiment analysis. Information, 15(8), p.473.

[16] Gao, L., 2024, February. Comparison of dqn and double dqn reinforcement learning algorithms for stock market prediction. In 2023 International Conference on Data Science, Advanced Algorithm and Intelligent Computing (DAI 2023) (pp. 169-177). Atlantis Press.

[17] Shi, Y., Li, W., Zhu, L., Guo, K. and Cambria, E., 2021. Stock trading rule discovery with double deep Q-network. Applied Soft Computing, 107, p.107320.

[18] Zhao, J., Zhang, C., Wang, C. and Yang, P., 2025. Learning from Expert Factors: Trajectory-level Reward Shaping for Formulaic Alpha Mining. arXiv preprint arXiv:2507.20263.

[19] Zia Uddin, A., 2024. Resilient Reinforcement Learning with Reward Shaping.

[20] Yuan, M., Li, B., Jin, X. and Zeng, W., 2023, July. Automatic intrinsic reward shaping for exploration in deep reinforcement learning. In International Conference on Machine Learning (pp. 40531-40554). PMLR.

[21] Rodinos, G., Nousi, P., Passalis, N. and Tefas, A., 2023, June. A Sharpe Ratio based reward scheme in Deep Reinforcement Learning for financial trading. In IFIP international conference on artificial intelligence applications and innovations (pp. 15-23). Cham: Springer Nature Switzerland.

[22] Kölle, M., Matheis, T., Altmann, P. and Schmid, K., 2023. Learning to participate through trading of reward shares. arXiv preprint arXiv:2301.07416.