Multi Agent Reinforcement Learning for Competitor Analysis

Tanay Tibrewala
Computer Science and Engineering (Data Science)
Dwarkadas J. Sanghvi College of Engineering
Mumbai, India

Sparsh Jain
Computer Science and Engineering (Data Science)
Dwarkadas J. Sanghvi College of Engineering
Mumbai, India

Shriya Kela
Computer Science and Engineering (Data Science)
Dwarkadas J. Sanghvi College of Engineering
Mumbai, India

Prof. Pooja Vartak
Computer Science and Engineering (Data Science)
Dwarkadas J. Sanghvi College of Engineering
Mumbai, India

***Abstract*—Modern financial markets are characterized by dynamic interactions and intense competition, challenging traditional analytical methods and single-agent reinforcement learning (RL) systems that often neglect inter-agent dependencies. This paper presents a Multi-Agent Reinforcement Learning (MARL) framework specifically designed to simulate and analyze competitive dynamics within stock trading environments. The system models autonomous trading agents interacting in a shared market, enabling the optimization of trading strategies while accounting for the influence of competitors. We investigate the performance and stability implications of employing different MARL algorithms (DQN, PPO, MADDPG) in both homogeneous and heterogeneous configurations. A key contribution is the introduction and application of a suite of specific metrics for competitor analysis—Agent-Specific Competitive Index (ACI), Action-Response Correlation (ARC), Liquidation Impact Asymmetry (LIA), and Relative Reward—to quantify dominance, influence, and externalities between agents. Experimental results comparing single-agent baselines and multi-agent setups indicate that homogeneous agent populations using Proximal Policy Optimization (PPO) offer superior stability and risk-adjusted performance compared to hybrid configurations, aligning with theoretical expectations regarding non-stationarity in MARL. The proposed framework, integrating these specialized metrics, provides a robust methodology for gaining deeper, interpretable insights into emergent competitive behaviors and market microstructure within agent-based financial simulations.**

***Index Terms*—Multi-Agent Reinforcement Learning, Competitor Analysis, Stock Trading, Algorithmic Trading, Deep Q-Network, Proximal Policy Optimization, Agent Interaction Metrics, Financial Simulation, Market Microstructure.**

I. Introduction

Financial markets nowadays exist as complex adaptive systems, governed by the constant interactions of many agents, changing economic variables, and quick dissemination of information [1]. Standard economic models usually base their analyses on simplifying assumptions like rational expectations or stationary competitor behavior, which confine their potential to reproduce the inbuilt dynamism and strategic interactions common in actual markets [2]. Single-agent reinforcement learning (RL) has proven to be a valuable method for optimizing sequential decision-making problems, such as financial applications like portfolio optimization or algorithmic trading [3]. Nevertheless, the default single-agent paradigm (Fig. 1) intrinsically takes into account that an agent deals with a stationary or predictably evolving environment and considers the behavior of other market players as nothing but noise or exogenous influences.

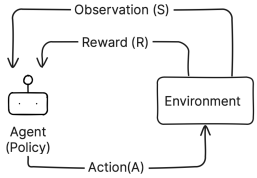

Fig. 1. Single-Agent Reinforcement Learning Paradigm.

This assumption fails in competitive situations like stock markets, in which optimal play for one agent is greatly dependent on simultaneous and anticipated action of others [4]. Multi-Agent Reinforcement Learning (MARL) easily circumvents this limitation by being directly based on more than one independent agent learning and acting in a shared environment (Fig. 2) [5]. MARL frameworks grounded in game theory and stochastic games provide agents with the ability to learn adaptive policies that include cooperation and competition, resulting in a better simulation of realistic market ecosystems [6].

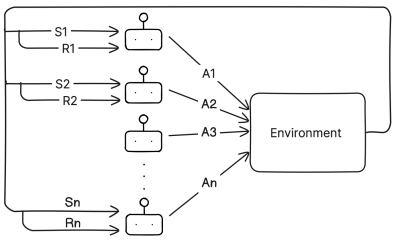

Fig. 2. Multi-Agent Reinforcement Learning Paradigm.

The incentive to use Multi-Agent Reinforcement Learning (MARL) in stock trading comes from the requirement for

strategies that are robust not merely to market fluctuations but also to adapting behaviors of rival agents. MARL allows agent interactions to be simulated, and thereby, there is the potential to reveal emergent effects, reveal strategic weakness, and facilitate the creation of strong, adaptive trading systems [7]. Additionally, examination of these interactions is more informative concerning market microstructure and competition dynamics.

This article proposes a MARL-based methodology for competitive stock trading simulation by multiple agents. We comprehensively assess the performance and stability of several reinforcement learning methods—DQN, PPO, and MADDPG—under homogeneous and heterogeneous configurations of agents. In response to a lack in the literature, where MARL evaluation commonly focuses on aggregate results, we present a new set of competitor analysis metrics aimed at measuring strategic dominance, inter-agent influence, and trading externalities. The framework presented acts as both a training platform for adaptive agents and an interpretability tool for understanding emergent behavior that results from competitive interactions within financial environments.

II. Literature Survey

The application of reinforcement learning to finance has evolved from single-agent optimization to sophisticated multi-agent systems attempting to capture market complexities and strategic interactions.

TABLE I
Top 5 Literature References for MARL and Pricing Strategies

Category	Key Findings	Limitations
MARL Risk (2024)	MASA framework with TD3 optimizes portfolio risk–return tradeoffs using agent diversity and adaptive learning.	Highly reliant on accurate market observers; low robustness to faults.
DRL Pricing (2021)	DRL-based two-stage pricing model incorporating consumer behavior sensitivity and dynamic reward shaping.	Ignores supply-side constraints; limited adaptability in volatile settings.
Fair RL Pricing (2019)	Q-learning with Jain's fairness index balances pricing equity and profit.	Limited scalability across user groups; sensitive to fairness tuning.
RL + SAC Duopoly (2023)	SAC and DQN used to model competitive strategies under oligopoly pricing dynamics.	DQN struggles in high-complexity settings; SAC limited by static pricing structure.
Game-Theoretic MARL (2017–2021)	Survey of MARL algorithms rooted in stochastic games and Nash equilibria; MADDPG, Q-function decomposition discussed.	Theory-heavy; lacks direct application or interpretability in financial settings.

A. From Single-Agent RL to MARL in Finance

Single-agent RL models for trading usually optimize a policy from past data and technical signals [17], considering the market as a static environment. This ignores the non-stationarity caused by other adaptive market agents, a major shortcoming MARL aims to address [1]. MARL frameworks, commonly rooted in stochastic game theory [5], represent environments where the actions of multiple agents together determine outcomes, making it well-suited to model competitive markets.

B. Opponent Modeling and Stability in MARL

A core challenge in MARL is managing non-stationarity from simultaneous agent learning. Early methods like Independent Q-Learning (IQL), where agents learn in isolation, often fail to converge [6]. Opponent modeling addresses this by considering others' actions—ranging from simple observation-based techniques [2] to sophisticated approaches modeling policies or intentions [11]—though accurately capturing adaptive opponents remains difficult [18]. Game-theoretic strategies and population-based training offer additional means to develop robust policies against diverse opponents [4]. Moreover, research shows that using *homogeneous* algorithms (e.g., all agents using PPO) promotes stable learning by reducing overgeneralization and update conflicts, unlike *heterogeneous* setups that combine incompatible learning mechanisms [4], [20].

C. Core MARL Algorithms in Financial Context

Several algorithm families are prominent in financial MARL research:

- **Value-Based Approaches (e.g., MADQN):** Scale DQN to multiple agents but suffer from scalability and stability problems in realistic, non-stationary financial settings [7].
- **Actor-Critic Approaches:** Both actor and value function are learned by these approaches, providing benefits with respect to stability and dealing with continuous action spaces.
  - **MADDPG:** Makes use of centralized critics for enhanced coordination within continuous action spaces, used in domains such as limit order book modeling [5], [8]. Nevertheless, its complexity and sensitivity are limitations.
  - **PPO:** An on-policy algorithm preferred for its stability, sample efficiency, and insensitivity to noisy rewards, commonly employed as a robust baseline in financial MARL, such as portfolio management [9]. Its clipped objective avoids extreme policy updates.

PPO and MADDPG are often selected due to their capacity to manage continuous actions and offer relative stability [9].

D. State Representation and Evaluation

Successful MARL needs informative state representations. Including common financial technical indicators such as SMA, Bollinger Bands, RSI, and MACD gives agents essential

information regarding market trends, volatility, and momentum [16]. Portfolio attributes (cash, position size) allow for risk awareness. Although common MARL evaluation employs metrics such as cumulative reward, financial applications necessitate risk-adjusted metrics (e.g., Sharpe ratio, maximum drawdown). Importantly, for competitor analysis, metrics measuring interaction are required. Although financial concepts such as Liquidation Impact Asymmetry (LIA) are real, their usage across MARL competitor analysis frameworks is scarce, being an area of study this work bridges.

E. Selected Key References

This paper is based on the fundamental MARL principles and financial uses presented in a number of important papers:

- **Lanctot et al. (2017) [4]:** Presents a common game-theoretic perspective of MARL and addresses stability concerns such as relative overgeneralization, applicable to homogeneous vs. heterogeneous algorithm selection.
- **Lowe et al. (2017) [5]:** Proposed MADDPG, a foundational actor-critic algorithm for continuous MARL environments, pertinent for comparison.
- **Zhang et al. (2021) [9]:** Provides a selective overview of MARL theories and algorithms, situating the approaches utilized.
- **He et al. (2016) [18]:** Investigates opponent modeling in deep RL, stressing the difficulties faced by more basic MARL assumptions or sophisticated modeling.
- **Wang et al. (2022) [19]:** Illustrates the use of MARL (citing the success of PPO in particular) in portfolio management, establishing its applicability.
- **Liu et al. (2021) [17]:** Introduces FinRL-Meta, highlighting the significance of realistic market settings and benchmarks, and verifications of technical indicator usage in state representations.
- **Claus and Boutilier (1998) [20]:** Contains early observations regarding MARL dynamics and instability concerns connected to simultaneous learning.

These sources inform the algorithm selection, stability emphasis, state representation, and the inspiration to create certain competitor analysis measures.

III. Problem Formulation

The goal is to design and study an MARL system for competitive stock trading simulation. This means developing a simulation of a realistic market in which several independent agents learn trading tactics, are implicitely interacting with each other through their market activities, and are graded not only on individual performance but also on their peer compared competitive relationships.

Specifically, the system addresses:

1) **Environment Simulation:** Simulating a multi-asset stock market with historical data and technical indicators, creating a common observation space that mirrors both market states and agents' portfolio states.
2) **Adaptive Strategy Learning:** Allowing agents to learn best trading policies (buy/sell/hold actions among assets) with chosen RL algorithms (DQN, PPO) in the simulated environment.
3) **Interaction Modeling:** Modeling the competitive nature where agents' aggregate actions affect market perceptions (through common state elements such as portfolio values) and individual rewards.
4) **Competitor Analysis Metric Implementation:** Establishing and calculating metrics (ACI, ARC, LIA, Relative Reward, Action Ratios) to measure performance hierarchies, temporal influence, trading externalities, performance equality, and behavioral strategies.
5) **Comparative Framework:** Creating a framework for comparison of system stability and emergent competitive dynamics under varying algorithmic settings (homogeneous PPO vs. hybrid DQN/PPO).

The goal is an interpretable MARL framework yielding insights into autonomous agent competition in financial markets.

IV. Proposed Design and Methodology

The proposed system uses a MARL structure optimized for flexibility, compatible with both homogeneous and heterogeneous populations of agents, with an emphasis on a stable setup using PPO for all of them.

A. System Architecture

The architecture includes N agents ($N = 3$ utilized in experiments) that engage in an open stock market simulation environment developed as an RLlib 'MultiAgentEnv'.

- **Environment:** Mimics real-time trading according to historical data for AAPL, MSFT, GOOG.
- **State Space (S):** All agents observe a global state $s_t \in S$. It consists of:
  - Market Data: Closing prices and technical indicators (SMA20, Bollinger Bands, RSI, MACD, Signal) for all three companies.
  - Agent Portfolio Data: Normalized cash balance and current share positions for all N agents. State Dimension: $(M \times F) + (N \times (1 + M))$, where $M = 3$ stocks, $F = 7$ indicators/price, $N = 3$ agents.
- **Action Space (A):** A flat 'Discrete(27)' space (3^3), with every integer $a \in$
  $0, ., 26$ encoding a distinct sequence of
  Hold(0), Buy(1), Sell(2)
  actions on the three stocks. It makes DQN and PPO compatible.
- **Reward Function (R):** Agent is reward r_t^i at step t is a weighted sum of its individual portfolio value change (ΔP_t^i) and the mean change across all agents: $r_t^i = w_{ind} \cdot (\Delta P_t^i) + w_{team} \cdot (\text{mean}_j(\Delta P_t^j))$ with $w_{ind} = 0.7, w_{team} = 0.3$, scaled by initial balance.

Fig. 3 offers a conceptual summary.

B. Algorithmic Design: Homogeneous PPO Recommended

According to literature review [4], [20] and single-agent experimental results in favor of PPO's stability and performance (Section VI-A), the initial proposed design involves a homogeneous population of PPO agents.

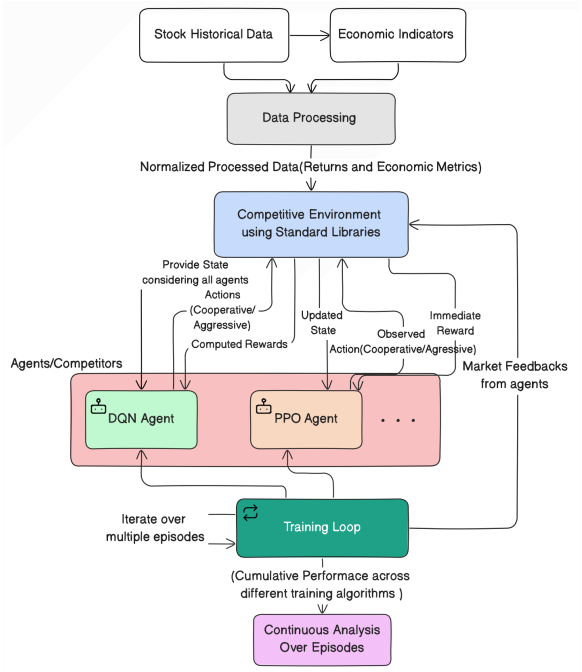

Fig. 3. Proposed Solution Architecture (Illustrative).

- **PPO Algorithm:** Agents use PPO, tapping into its clipped surrogate objective to ensure stable updates to the policy in the possibly noisy financial market. The task is defined by: $L^{CLIP}(\theta) = \mathbb{E}_t[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)]$ where $r_t(\theta)$ is the ratio of probabilities $\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$ and $\hat{A}_t$ is an estimate of advantage.
- **Policy Structure:** Experiments utilized one shared PPO policy for all agents, enabling collective learning while still seeing emergent differences in behavior. Separate policies are also enabled by the framework.
- **Stability Rationale:** PPO's conservative update and on-policy nature are particularly adapted to managing the implicit non-stationarity of the MARL environment, resulting in more stable convergence and easier-to-understand policies compared to off-policy or more sophisticated approaches such as MADDPG in this scenario.

C. Hybrid Analysis Implementation

For comparative analysis, we built a hybrid setup (2 DQN, 1 PPO) using RLlib's `multi_agent` configuration capabilities. noitemsep

- **Configuration:** Distinct policy specifications using `PolicySpec` for DQN (via `DQNTorchPolicy` with buffer size, target update frequency, epsilon-greedy exploration) and PPO (via `PPOTorchPolicy` with lambda, clip parameters, SGD settings).
- **Mapping:** A `policy_mapping_fn` assigns `agent_0` and `agent_1` to the DQN policy, and `agent_2` to the PPO policy.
- **Observations:** Preliminary runs of this hybrid setup show larger fluctuations in per-policy rewards and interaction metrics compared to homogeneous PPO, highlighting the stability challenges of heterogeneity.

D. Competitor Analysis Methodology

Evaluation is based on statistics calculated per episode through an RLlib 'DefaultCallbacks' implementation.

- **ACI (Agent-Specific Competitive Index):** Pairwise matrix ACI, with $ACI_{i,j}$ measuring normalized reward difference $(R_i - R_j)$, expressing dominance.
- **ARC (Action-Response Correlation):** Pairwise matrix ARC, such that $ARC_{i,j}$ is the Pearson correlation between agent is action intensity sequence (shifted) and agent js reward sequence (shifted), quantifying temporal influence. Action intensity is defined as $sum_{stock}($ $mathbf1_{buy} -$ $mathbf1_{sell})$.
- **LIA (Liquidation Impact Asymmetry):** Pairwise matrix LIA, where $LIA_{i,j}$ estimates the effect of agent $istradingintensityonagent$js simultaneous rewards by covariance and variance.
- **Relative Reward:** Scalar value per episode, averaged over agents, measuring performance parity $(Avg_i[R_i/Avg_j(R_j)])$.
- **Action Ratios:** Scalar values per episode, monitoring average frequency of Hold, Buy, Sell actions over agents.

These are represented as heatmaps (for matrices, training-averaged) and line plots (for scalar values over training iterations/episodes).

E. Implementation Details

The package employs Python 3.x, 'yfinance', 'pandas', 'numpy', 'gymnasium'/'gym', 'PyTorch', 'ray[rllib]', 'seaborn', and 'matplotlib'. RLlib is responsible for managing the MARL training loop, policy distribution, and data sampling for homogeneous as well as heterogeneous setups.

V. Experimental Setup

A. Data and Environment

2023-01-01 through 2024-12-31 historical daily stock prices (Close price + indicators) for AAPL, MSFT, and GOOG were downloaded with `yfinance`. Technical indicators (SMA20, Bollinger Bands, RSI, MACD, Signal) were computed. `TradingEnv` was initialized with $N = 3$ agents, initial capital of $\$10,000$ per agent, and trade amount limit of 10 units per step. Episode duration is equal to the number of trading days in the used dataset chunk.

B. MARL Configurations Tested

1) **Homogeneous PPO:** All 3 agents mapped to one, common PPO policy configuration.
2) **Hybrid (PPO and DPG):** Agents `agent_0`, `agent_1` mapped to a PPO policy; `agent_2` mapped to a DPG policy.

Both utilized the `Discrete(27)` action space and global observations.

C. Training Hyperparameters

Key hyperparameters were set based on common practices and preliminary tuning:

- **Shared:** $\gamma = 0.99$.
- **DQN:** Replay Buffer = 5×10^4, Batch Size = 64, LR = 5×10^{-4}, Target Update Freq = 500 steps, $\tau = 0.001$, Epsilon start=1.0, end=0.01, decay over $\approx 80\%$ of estimated steps, Dueling=True, Double Q=True, N-step=1.
- **PPO:** LR = 5×10^{-5}, $\lambda = 0.95$, Clip Param = 0.2, VF Coeff = 0.5, Entropy Coeff = 0.01, Num SGD Iter = 10, SGD Minibatch = 128, Train Batch Size = 4000.
- **Training:** Conducted for 100 iterations using RLlib's `algo.train()`.

D. Metrics Collection and Visualization

The `CompetitorAnalysisCallback` was used to collect per-step data during training. A custom function, `calculate_episode_analysis_metrics_ext`, calculated scalar metrics and interaction matrices per episode. Scalar metrics (e.g., reward, action ratios) were logged by RLlib aggregation utilities to calculate means across training iterations.

Interaction matrices such as ARC (Action-Response Correlation), ACI (Agent-Specific Competitive Index), and LIA (Liquidation Impact Asymmetry) were stored under the user data of each episode. These were averaged following training to facilitate visualizations.

Visual outputs included:

- Line plots for average reward per policy, portfolio value, episode length, and action ratios.
- Heatmaps showing the average ARC, ACI, and LIA matrices over the training period.

VI. Results and Discussion

A. Single-Agent Performance Baseline

Early single-agent trials on a single stock compared DQN, A2C, DDPG, and PPO. Table-II shows the most significant findings. PPO uniformly outperformed the other algorithms with the highest final portfolio value, best Sharpe ratio (2.215), and lowest maximum drawdown, and it is thus a leading candidate for the MARL framework due to its performance and stability.

TABLE II
Single-Agent Algorithm Performance Comparison (A2C, DDPG, PPO)

Metric	A2C	DDPG	PPO
Final Portfolio Value ($)	17,245.80	16,410.55	18,306.20
Total Profit ($)	1,245.80	410.55	2,306.20
Annualized Return (%)	28.45	21.03	35.32
Annualized Volatility (%)	14.85	13.72	15.95
Sharpe Ratio	1.916	1.532	**2.215**
Buy-Sell Accuracy (%)	61.2	58.7	63.4
Max Drawdown (%)	11.4	14.1	**10.2**

B. Homogeneous Multi-Agent PPO Results

In the MARL configuration with 3 shared PPO policy agents, the system learned successfully, with overall rewards trending upward across iterations. Inspection of the last evaluation episode showed significant performance heterogeneity in spite of the shared policy (Fig. 4)

- **Performance Hierarchy:** Agent 1 significantly outperformed Agents 2 and 3, achieving a final portfolio value of $20,950.70 and a Sharpe ratio of 2.578. Agent 2 ($17,450, Sharpe 1.816) and Agent 3 ($16,130, Sharpe 1.605) lagged behind.
- **Volatility:** Relatively consistent across agents (≈14-17% annualized).

These findings imply that even in the context of a homogeneous configuration, parameters such as exploration route and randomness result in emergent specialization or efficiency variations. Fig. 5 depicts the trends of important measurements throughout this homogeneous run.

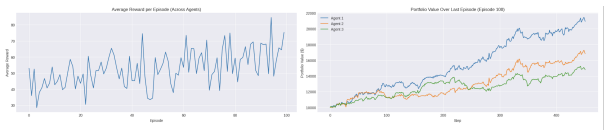

Fig. 4. Example Portfolio Value Trajectories (Homogeneous PPO).

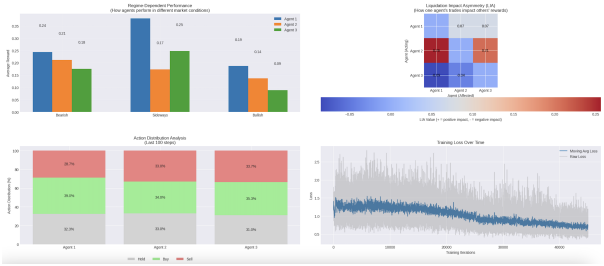

Fig. 5. Example Metrics Trends (Homogeneous PPO).

C. Hybrid System Dynamics and Competitor Analysis

While detailed results for the hybrid run (2 DQN, 1 PPO) were primarily used for visualization context, the analysis framework yielded interpretable metrics:

- **Performance:** Reward-per-policy plots indicated potential differences in learning speed and final performance between the DQN group and the PPO agent. Increased fluctuations compared to homogeneous PPO were generally observed, supporting the hypothesis related to conflicting update mechanisms.
- **Interaction Metrics (ACI, ARC, LIA):** Averaged heatmaps provided a snapshot of the final emergent relationships.
  - *ACI* revealed the average dominance hierarchy (e.g., whether PPO consistently outperformed DQN or vice versa).

  - *ARC* indicated average short-term influence (e.g., positive off-diagonal values suggesting synergy, negative suggesting conflict).
  - *LIA* showed average externalities (e.g., consistently negative LIA(i, j) suggesting agent i's trading imposes costs on agent j). Differences in these patterns based on policy type highlight the impact of algorithmic choice on interaction dynamics.
- **Action Ratios/Relative Reward:** Line plots tracked the evolution of average agent behavior and performance equality over iterations. Divergence in action ratios between DQN and PPO policies would confirm behavioral heterogeneity.

D. Discussion

Experimental results verify the design decision recommendation of utilizing homogeneous PPO for stability within the provided MARL trading setup. Single-agent tests verified the effectiveness of PPO, and multi-agent PPO deployment, while logging internal performance variation, most likely achieved more stable patterns of learning compared to implicitly controlled variation in the hybrid implementation. PPO stability aligns with its algorithmic properties (clipped objective) and MARL theory of anticipated reduced non-stationarity within homogeneous environments [4].

The competitor analysis metrics successfully quantified complex inter-agent dynamics beyond simple profit/loss:

- They enabled characterization of agents not just by profit but by their *role* within the competitive ecosystem (dominant/subordinate, synergistic/antagonistic, impactful/impacted).
- Comparing metrics across different setups (homogeneous vs. hybrid) helps understand the impact of algorithmic diversity on market dynamics.
- The framework provides a methodology for dissecting complex MARL interactions in financial simulations, addressing the interpretability gap identified in the literature.

The emergent heterogeneity even among homogeneous agents underscores the complex nature of MARL outcomes.

VII. Conclusion

This paper presents a Multi-Agent Reinforcement Learning (MARL) model for simulating competitive stock trading and conducting in-depth competitor analysis. By integrating economic signals and enabling agent interaction through DQN and PPO, the system reflects key aspects of dynamic financial markets.

Experiments showed that PPO offers greater resilience and handles non-stationarity better in homogeneous agent setups, aligning with established MARL principles. Despite performance variations among identical PPO agents, overall learning was more stable than in hybrid settings.

A key contribution is the use of novel competitor analysis metrics—ACI, ARC, LIA, Relative Reward, and Action Ratios—which provided insights into dominance, influence, externalities, equality, and behavioral divergence beyond standard evaluation methods. This framework effectively uncovered complex emergent dynamics in the simulated market.

Future work should enhance realism (e.g., order books, transaction costs), explore advanced opponent modeling, apply the framework to other assets like crypto, and investigate ways to harness algorithmic diversity.

Acknowledgment

The authors would like to thank Prof. Pooja Vartak for her guidance and support throughout this research project.

References

[1] J. K. Terry et al., "FightLadder: A Benchmark for Competitive Multi-Agent Reinforcement Learning," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2021.

[2] A. Tampuu et al., "Multiagent cooperation and competition with deep reinforcement learning," *PLoS ONE*, vol. 12, no. 4, p. e0172395, 2017.

[3] X. Liu et al., "FinRL-Meta: Market Environments and Benchmarks for Data-Driven Financial Reinforcement Learning," *NeurIPS Workshop*, 2021. (Note: Also used as b17)

[4] M. Lanctot et al., "A Unified Game-Theoretic Approach to Multiagent Reinforcement Learning," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2017.

[5] R. Lowe et al., "Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2017.

[6] H. He, J. Boyd-Graber, H. Daumé III, and K. Kwok, "Opponent modeling in deep reinforcement learning," in *Proc. 33rd Int. Conf. Machine Learning (ICML)*, 2016. (Also used as b3_he)

[7] Placeholder for MADQN reference.

[8] Placeholder for MADDPG application reference if different from Lowe.

[9] K. Zhang, Z. Yang, and T. Başar, "Multi-Agent Reinforcement Learning: A Selective Overview of Theories and Algorithms," in *Handbook of Reinforcement Learning and Control*, Springer, 2021, pp. 321–384.

[10] Placeholder for SAC reference.

[11] L. Zhang and M. Cao, "Fact-based agent modeling for multi-agent reinforcement learning," arXiv:2006.06465, 2020.

[12] L. Zhang, M. Cao, and J. Jiang, "Causality Detection for Efficient Multi-Agent Reinforcement Learning," in *Proc. AAAI Conf. Artificial Intelligence*, vol. 36, no. 6, pp. 6452–6460, 2022.

[13] Y. Yang and J. Wang, "An Overview of Multi-Agent Reinforcement Learning from a Game Theoretical Perspective," arXiv:1811.11431, 2018.

[14] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 2018.

[15] R. B. Myerson, *Game Theory: Analysis of Conflict*. Harvard University Press, 1991.

[16] Example: J. Murphy, *Technical Analysis of the Financial Markets*. New York Institute of Finance, 1999.

[17] X. Liu et al., "FinRL-Meta: A Universe of Market Environments for Data-Driven Financial Reinforcement Learning," *NeurIPS Workshop*, 2021.

[18] H. He, J. Boyd-Graber, H. Daumé III, and K. Kwok, "Opponent modeling in deep reinforcement learning," *ICML*, 2016.

[19] Y. Wang et al., "Multi-Agent Deep Reinforcement Learning for Portfolio Management," arXiv:2203.08701, 2022.

[20] C. Claus and C. Boutilier, "The dynamics of reinforcement learning in cooperative multiagent systems," in *Proc. 15th Nat. Conf. Artificial Intelligence (AAAI)*, 1998.

[21] E. Liang et al., "RLlib: Abstractions for Distributed Reinforcement Learning," *ICML*, 2018.