Exploring Effective Market Entry Strategies with Reinforcement Learning

Eric Thomas, Christopher Archibald, Stephen Sorensen, and David Bryce

Submitted to *Applied Artificial Intelligence*

May 6, 2026

Abstract

Entering new product markets is a primary growth mechanism for many firms. However, choosing the sequence and timing of market entrances is a nontrivial problem for which artificial intelligence (AI)-based approaches remain underexplored. The complexities of these market entry decisions include uncertainty over future market demands and costs as well as the impact on each firm of the decisions of the potentially many other firms present. These decisions are complex enough that a complete theoretical analysis of the situation is infeasible, suggesting the possibility of using other techniques, like AI. In this work, we introduce the Market Entry Game (MEG) to model market entry decision-making and apply existing reinforcement learning (RL) techniques to the problem of determining effective strategies for this game. Using a MEG simulator, we show that RL-based agents outperform rule-based agents in their ability to maximize capital through their market-entry decisions. Our results show that AI can make effective market entry decisions and uncover novel insights in strategic management.

Keywords: Market entry; reinforcement learning; agent-based computational economics; strategic management

I. Introduction

Markets are differentiated concentrations of buyers and sellers exchanging goods and services for value. Markets often exist in clusters, where the know-how required for companies to operate in one market may be related to or shareable between the know-how needed in another. Specifically, capabilities—which consist of complex sets of internal firm processes, human knowledge, machines and equipment, and other resources—are developed over time as a firm operates in a market. Some of these capabilities are extensible to other markets. For example, Amazon's dominance in the e-commerce market endowed the company with capabilities that then facilitated its successful entry into the cloud computing market. This was in large part due to the common production requirements of the two markets, such as server management, software engineering talent, infrastructure scalability, and large-scale data analytics. Conversely, Amazon's attempts to enter unrelated markets, such as smartphones with the Fire Phone, were unsuccessful because those markets required

Eric Thomas (Corresponding author; eht23@byu.edu; ORCID: 0009-0000-3729-3140) and Christopher Archibald (archibald@cs.byu.edu; ORCID: 0000-0001-7258-3805) are with the Department of Computer Science at Brigham Young University.

Stephen Sorensen (soren92@byu.edu) and David Bryce (dbryce@byu.edu) are with the Department of Management in the Marriott School of Business at Brigham Young University.

capabilities the company lacked, including advanced hardware design expertise, telecommunications carrier relationships, and deep experience in consumer device ecosystems.

The capacity for a firm to extend its capabilities to other markets is a key driver of firm growth and expansion. New market entry provides the firm with growth avenues resulting in total revenue increases and, where such efforts are profitable, contributions to a firm's capital stock. Similarly, firms divest themselves of markets when they find that performance is suffering or they no longer possess a rationale for owning production capability in that market. Since there exists a systematic relatedness structure between the resources required to compete in pairs of markets, firms may engage in strategic behavior to improve their performance. Specifically, firms may choose to enter markets that are related to those they already hold in their portfolios in order to access economies of scope in production and other complementarities, such as those available by combining common consumer preferences across markets. Firms that enter related markets ahead of rivals or find portfolio configurations that are superior to those held by rivals can achieve above-normal economic returns.

These game dynamics characterize the real-world behavior among diversifying firms. When and how should firms exploit markets related to those in which they already operate? When should they explore entirely different areas of the economy? Complicating these decisions is the fact that numerous other factors influence success in ways that are not fully known or predictable. For example, actual realized costs, expected entry and exit behavior of other firms, and actual available profits may be estimable but cannot be known for certain. Therefore, boundedly rational firms must make market entry choices under uncertainty. To do so, they may develop a set of simple rules [1] or heuristics to overcome the burden of complexity.

This paper proposes a model that enables the study of strategic behavior by firms making market entry choices under uncertainty. Specifically, we develop a market simulation model that permits tests of whether AI firm agents can outperform heuristic or rule-based firm agents, and if so, how they do it. The present effort builds the model and illustrates promising results for future research. We raise the following research question as the focus of this paper: *Can artificial intelligence (AI) discover effective market entry and exit strategies?* We simulate an economy that incorporates numerous representations of real-world factors that affect success and apply existing reinforcement learning (RL) techniques to uncover effective market entry and exit strategies. Our results demonstrate that RL-based agents outperform rule-based agents in their ability to maximize capital through market entry and exit decisions. These initial findings serve as proof of concept that modern AI can uncover novel insights in the field of strategic management. Furthermore, while our simulator, which we make publicly available, may not account for every economic complexity, it provides a robust platform upon which researchers can build to model their specific scenarios.

The contributions of this paper are as follows:

1) We formalize the problem of market entry decision-making as the *Market Entry Game* (MEG).
2) We introduce a simulator which implements MEGs in a way that supports application of (RL) algorithms.
3) We apply RL to a variety of simulated MEGs and show that the learned policies can outperform both naïve and sophisticated heuristic agents.

These contributions lay the groundwork for further principled application of AI techniques, including RL, to market entry problems. The potential insights gained could help

actual firms make better market entry decisions.

The paper proceeds as follows: Section II outlines the related literature on market diversification, AI-driven game strategy, agent-based computational economics, and business applications for AI. In Section III, we formulate the market diversification problem that is the focus of our work. Section IV then describes our simulator implementation of this problem in terms of its economic characteristics and its mechanics. Section V provides background on RL and the specific algorithms we employ, as well as details regarding the interactions between our simulator and off-the-shelf implementations of these algorithms. Section VI details how we train and evaluate our agents and offers insights we glean from the RL agents' emergent behavior. Section VII concludes and highlights opportunities for future work.

II. Literature Review

A. Market Diversification

The study of the performance effects of corporate diversification into related and unrelated markets has a long history in the strategic management literature. In an early landmark study [2], Rumelt showed that firms diversifying into related markets outperformed those diversifying into unrelated markets. Much follow-on research confirmed this basic finding. The predominant reason for the main result is resource sharing between pairs of businesses inside corporate portfolios. Specifically, economies of scope arise in the combination of production activities inside the same firm when those activities draw on common know-how, processes, assets, or other resources [3], [4]. When such resources have some excess capacity in use or otherwise possess a non-rivalrous character (not consumed in use), additional activities can be added without incurring the full incremental cost of production. This efficiency leads to performance enhancement under related diversification.

Research applying AI and machine learning to corporate diversification strategy is still nascent, and this study is an early step toward identifying the insights and benefits these tools can bring to strategic management.

B. Agent-Based Computational Economics

Agent-Based Computational Economics (ACE) is "the computational study of economic processes modeled as dynamic systems of interacting agents" [5]. ACE is a bottom-up approach to studying dynamic economic systems where outcomes may be intractable or computationally infeasible using traditional analytical methods. Using ACE, researchers create individual economic agents that make decisions according to individual incentives or rewards. The agents are introduced into an economic environment with initial conditions chosen by the researchers. Researchers watch for emergent agent behavior, making no intervention in the economic environment once the simulation has begun.

One application area for ACE is that of optimal tax policy. In a recent study [6], researchers at Salesforce and Harvard employed a deep RL approach to determine a tax schedule that maximizes a social welfare function accounting for equality and productivity. The researchers modeled both the individual economic actors as well as the policy-setting government as RL agents, resulting in dynamic tax policies that are robust in the face of tax-gaming strategies. Interestingly, the optimal tax schedule was found to be qualitatively different from both the actual U.S. tax schedule and the notable Saez tax framework in that it is U-shaped—a both progressive and regressive tax schedule that rewards taxpayers for moving closer to the middle of the income distribution.

Meanwhile, researchers at DeepMind studied emergent bartering behavior in a complex environment where agents have varying preferences and production abilities and goods vary in abundance by location [7]. The research serves as both a computational confirmation of basic microeconomic principles as well as an environment in which new AI techniques can be explored for the advancement of AI itself.

Earlier studies also used ACE techniques to analyze emergent population-level economic phenomena; however, these studies predate recent breakthroughs in deep learning and thus make primary use of rule-based agents without incorporating AI. We refer the interested reader to Section 2.5 of [7] for an excellent overview of this work.

C. Business Applications for AI

Market entry and exit strategy is one of numerous business problems where AI can be applied. The last decade has seen an explosion of industry interest in AI, with research from Goldman Sachs showing that mentions of AI in Russell 3000 earnings calls increased more than 23 times from 2015 to 2023 [8]. Here, we highlight some notable applications of AI to business problems.

- **Dynamic pricing:** Many e-commerce and travel websites employ dynamic pricing models that incorporate customer and competitor data to determine prices in real time [9].
- **Customer relationship management (CRM):** Applications of AI to CRM have been studied extensively, particularly in the areas of one-to-one marketing and loyalty programs [10].
- **Financial fraud detection (FFD):** Both the private and public sector use AI to sift through large amounts of data to detect fraudulent activity. In particular, AI is used extensively to identify insurance, corporate, and credit card fraud [11].
- **Talent acquisition:** Many companies use AI to assist in the hiring process [12]. Multinational corporations in particular receive a high number of candidates per job opening and benefit greatly from AI-assisted identification, attraction, and onboarding of new employees [13].
- **Asset trading:** Advanced AI models rapidly detect patterns in economic data and perform high-frequency trades [14]. Researchers have developed models for various purposes, including price prediction, portfolio management, stock selection, hedging strategy, and risk management [15].

Our work contributes to the literature on business applications for AI by providing a platform on which researchers can model their strategic scenario to gain insight about superior market entry strategies in their setting. While the economic model we employ in this work will require additional sophistication and realism before deployment in the real-world, it provides a crucial first step toward this novel application of AI.

D. AI-driven Game Strategy

Most of the early applications of AI to games focused on classic board games, such as checkers and chess [16]. These efforts led to IBM's monumental breakthrough in 1997 in which their Deep Blue program became the first AI model to defeat a reigning world chess champion [17]. The field of AI-driven game strategy has grown significantly since then, particularly in the last decade in which advances in deep learning have led to major breakthroughs. In 2016, for example, a research team led by Google DeepMind combined deep RL techniques with Monte Carlo simulation to develop a model that boasted a 99.8%

win rate against other Go programs and defeated the human European Go champion by five games to zero [18]. The next year, DeepMind released AlphaZero, which, provided no knowledge other than the game rules, progressed from random play to superhuman capability with 24 hours of training in Chess, Shogi (Japanese chess), and Go [19].

In 2019, DeepMind partnered with Team Liquid to release AlphaStar, an AI model created for the video game StarCraft II. This game presents a more difficult challenge than classic board games given that it is a multi-agent problem, the current state of the game is only partially observable, the state and action spaces are high-dimensional, and the game requires long-term planning over thousands of time steps [20]. Employing a variety of deep RL techniques, AlphaStar managed to outperform 99.8% of officially ranked human players [21].

Recent years have seen AI models succeed in a variety of game domains, including Poker [22], Dota 2 [23], and classic Atari video games [24]. Progress in games is significant because the techniques developed in these settings often transfer to problems of real-world consequence. For example, RL was first applied to the game of checkers [25] and now has broad applicability in robotics [26] and finance [27].

Modeling economic problems as games can similarly make them more tractable and provide insight into both the economic question at hand and the study of AI-driven game strategy itself. In this spirit, the Market Entry Game we introduce in Section III bears similarities to the classic board game Monopoly: players are endowed with initial capital, face investment opportunities, and exit the game upon bankruptcy. Unlike Monopoly, however, our model is designed to plausibly reflect real-world economic conditions, explicitly modeling supply and demand, fixed and variable costs, prices, and quantities. Importantly, whereas most game AI systems are designed for a small, fixed number of players, our framework supports an arbitrary number of agents, enabling the study of strategic behavior in larger and more economically realistic multi-agent environments.

III. Problem Formulation

A. Market Entry Game Definition

In this paper, we explore the efficacy of applying AI techniques to yield insights to market entry and diversification decisions faced by firms. One of the key contributions of this work is the formulation of this problem as a multi-agent stochastic game. We formulate this precisely as the *Market Entry Game* (MEG) (F, M, S, s_0, T, ρ, P, R, γ) where:

- $F = \{f_0, f_1, \ldots, f_n\}$ is the set of n firms participating in the MEG.
- $M = \{m_0, m_1, \ldots, m_k\}$ is the set of k markets. These markets implicitly determine the set of actions available to the firms, as during each turn, a firm can either 1) enter a market it isn't in, 2) exit a market it is in, or 3) do nothing, which will be indicated by the $\emptyset$ action.
- S is the set of states of the MEG. In a MEG, each state must, at a minimum, specify the amount of capital held by each firm, as well as the market presence (in/out) for each firm and market combination. $s_0 \in S$ is the initial state of the simulation.
- $T \subseteq S$ is the set of terminal states of the MEG. If these states are reached then the game ends.
- $\rho : S \setminus T \mapsto F$ is the turn function, which specifies for each non-terminal state the firm that is acting in that state.
- $P : (S \setminus T) \times (M \cup \emptyset) \mapsto \Delta(S)$ is the transition function, where $\Delta(S)$ is the set of all probability distributions over S. P specifies a specific distribution over next states for any state and market entry action chosen by the acting firm.

- $R = (r_0, r_1, \ldots, r_n)$, where each $r_i : S \times S \mapsto \mathcal{R}$ is the reward function for firm i, specifying the net change in capital for a realized transition between states.
- $\gamma \in [0, 1]$ is the discount factor.

The goal of an agent playing a MEG is to maximize the cumulative discounted reward of the firm(s) they control. (In our experiments, each agent controls exactly one firm, so the agent-firm distinction is unimportant.) Each agent's policy π will specify a distribution over actions to take for any state. The game begins in an initial state, s_0, which in this study is a state in which all firms have the same amount of starting capital and do not yet participate in any markets. At each step within the game, $\rho(s)$ determines which firm's turn it is to act. The acting firm chooses to enter a market, exit a market, or do nothing. The resulting change in state is used to determine a reward for each firm according to R. The simulation ends when a terminal state is reached, which in our implementation occurs when a maximum number of time steps have passed, or when all firms have gone bankrupt.

B. Discussion

While we model the MEG with firms taking turns entering and exiting markets, the game could alternatively be formulated using a simultaneous-move model in which all firms act at each time step and update their market portfolios synchronously. Both formulations may be appropriate in different real-world settings, depending on how quickly firms can enter and exit markets in a given industry. For example, firms entering markets within the automotive industry (e.g., battery-electric vehicles, hybrid SUVs, or autonomous driving platforms) typically have substantial time to observe and react to competitors' decisions, as entry opportunities are constrained by long product development and manufacturing setup cycles. Conversely, small restaurants within a neighborhood can update their menus rapidly, allowing them to enter or exit product markets on time scales as short as a single day. In this paper, we focus on sequential MEGs and leave the study of simultaneous MEGs to future work.

With the problem formulated in this manner, we can apply AI techniques that have proven effective in other game settings, as discussed in Section II-D. Applying these techniques requires a concrete implementation of the MEG that specifies all model components in a principled manner. In the next section, we describe the implementation details of our simulator, which serves as the environment in which AI agents learn effective market entry strategies.

IV. The Market Entry Game Simulator

Our MEG simulator consists of markets, firms, and agents. A market is a mechanism whereby buyers and sellers of a specific product engage in exchange. Each market is defined by a linear demand curve and a set of capabilities required for sellers to participate in that market. A firm is an entity that can choose to enter and exit markets. When one or more firms are in a market, they choose their production quantities according to their production policy (which, for this paper, is the Cournot oligopoly model) and the market price is then determined by the market demand curve. Firms are endowed with some starting amount of capital and that capital increases via revenue and decreases via market entry costs, fixed costs, and variable costs. An agent is an entity that seeks to maximize a firm's capital by controlling its market entry and exit decisions. Agents may be rule-based or AI-powered.

In the rest of this section, we provide more simulator implementation details and explain how our simulator instantiates the MEG defined in Section III. We proceed in three steps:

(1) we map each simulator component to the corresponding MEG element, (2) we describe how we model economies containing markets with varying degrees of overlap in terms of their requisite production capabilities, and (3) we specify how all quantitative variables (costs, demand, prices, profits) are computed.

A. Instantiating the Market Entry Game

Our simulator is an instantiation of the MEG tuple $(F, M, S, s_0, T, \rho, P, R, \gamma)$ as follows:

- **Firms** (F): Each simulation contains n firms $F = \{f_0, \ldots, f_n\}$, each controlled by exactly one agent.
- **Markets** (M): The economy contains k markets $M = \{m_0, \ldots, m_k\}$. Each market is defined by (i) a linear demand curve and (ii) a set of required capabilities (see Section IV-B).
- **States** (S): A state encodes each firm's current capital and each firm's market portfolio (in/out of each market). In addition, the state includes fixed economic parameters (e.g., demand curve parameters and market overlap structure) that remain constant within a simulation. In Section V-C, we explicitly list all state components that we provide to the AI agent as its state observation.
- **Initial state** (s_0): All firms begin with the same starting capital and empty market portfolios.
- **Terminal states** (T): A simulation terminates when a maximum number of time steps is reached or when all firms have gone bankrupt. In this study, all simulations last for a maximum of 100 macro time steps. We leave variable-max-length simulations as future work.
- **Turn function** (ρ): Each simulation proceeds over a sequence of macro time steps, each of which consists of multiple micro time steps. We first describe the events that occur during a single micro time step and then explain how micro time steps are aggregated into a macro time step.
  - *Micro time steps:* During a micro time step, at most one agent is designated as the acting agent. The acting agent may choose to enter any market it is not currently in, exit any market it is currently in, or do nothing. The market portfolios of all other firms remain fixed during that micro time step. After the acting agent makes its entry, exit, or do-nothing decision, production quantities are computed in each market, prices are determined, profits are calculated for each firm–market pair, and each firm's capital is adjusted accordingly.
  - *Macro time steps:* A macro time step consists of a sequence of micro time steps in which each agent is given exactly one opportunity to act. The simulator may also include a specified number of skip turns (micro time steps during which no agent may enter or exit a market) within each macro time step. In addition, the order of agent turns (including skip turns) can be randomized within each macro time step.
  - *Example:* Consider a simulation with 100 macro time steps, two agents, and two skip turns per macro time step. Each macro time step therefore contains four micro time steps, yielding 400 micro time steps in total. If turn order is randomized within each macro time step, one possible sequence of turns is
    $$\{(1, \text{skip}, \text{skip}, 2),\ (1, 2, \text{skip}, \text{skip}),\ (\text{skip}, 2, \text{skip}, 1),\ \ldots\}.$$
  - *Bankruptcy:* If a firm's capital falls below zero at any point during the simulation, the firm is declared bankrupt. It is removed from all markets and no longer

participates in the simulation. Bankruptcy is the terminal and worst possible outcome for a firm. Anytime a bankrupt firm is designated as the acting firm during a micro time step, that firm's turn is replaced with a skip turn.

- **Transition function** (P): Given the current state and the acting firm's action (enter, exit, or do-nothing), the next-state distribution is induced by (i) the portfolio update and (ii) the resulting market competition outcomes (Cournot quantities and demand-based prices) that update all firms' capital. Stochasticity stems from the randomization over turn order: because agent turn order is shuffled within each macro time step and because there are skip turns mixed in with this randomization, multiple next-states could result from any agent action.
- **Rewards** (R): Each firm's reward for a realized state transition is its net change in capital over that transition.
- **Discount factor** (γ): We rely on the built-in discount factor in the RL algorithms we employ to discount rewards. While necessary for RL algorithms to converge, this discount factor also has an intuitive economic interpretation: capital increases that occur sooner are valued more than those that occur later.

B. Market Entry Decisions and Market Similarity

As mentioned in the introduction, markets exist in related and overlapping clusters, where the materials, machinery, personnel, and processes required to supply goods and services in one market may be correlated with those needed in another. To model this abstractly, we conceptualize a large number of capabilities that exist in the economy, with each market requiring a specific subset of those capabilities. For example, a typical economy in our simulator might contain 2000 capabilities while each individual market within the economy requires a firm to possess a subset of 100 specific capabilities to produce products in that market. We index these abstract capabilities using the natural numbers $\mathbb{N}$, with the only significance of the indices being that index values closer together can be thought of as closely related capabilities (e.g., capabilities 203 and 204 represent more closely related capabilities than 106 and 805). Similarity between product markets is then modeled by how many requisite capabilities a given pair of markets have in common, as well as how *shareable* those capabilities are[1] [28].

In the simulator settings, we select how many markets exist in the economy and how related each is to the others. We achieve this by introducing the notion of a cluster: a group of markets whose requisite capabilities are drawn from the same underlying distribution. These clusters appear in the real world as *verticals*—sets of markets using common inputs and producing common outputs (e.g., retailing, financial services, and manufacturing). These clusters can be thought of as connections of markets, creating a hierarchical structure for markets.

[1] *Shareability* is the degree to which a firm benefits from a capability being required by multiple markets in its portfolio. For each capability, we introduce a shareability parameter s. For each market that requires some capability in which the firm participates, we let that capability's contribution to entry costs for that firm-market combination be $(1 - s)c + s(c/n)$, where s is the shareability parameter, c is the cost of the capability in question, and n is the number of markets in the firm's portfolio currently sharing that capability. This has the effect that when s is 0, a capability is not shareable at all and firms must pay c for every instance in which they use a capability; when s is 1, a capability is perfectly shareable and its cost gets split evenly between markets in which it is used; and intermediate values of s result in a mixed effect between these two extremes. Note that fixed costs and exit costs per market are simply a percentage of entry costs, so shareability affects these costs indirectly. We experimented with several different shareability levels but found that this did not lead to interesting results; thus, for all experiments in this paper, we use a shareability level of 0.5.

For each cluster, we select a mean and standard deviation. Using these parameters, capabilities are drawn and assigned to each market within a cluster. For example, suppose an economy contains 2000 capabilities and each market within the economy requires 100 capabilities. Cluster A is defined to have a mean of 500 and a standard deviation of 50. Therefore, for markets in cluster A, it is highly likely that capabilities 499, 500, and 501 will be selected as part of the market's required capabilities and less likely that capabilities 10 and 1407 will be selected. Note that capabilities are selected without replacement, so each market is guaranteed to have the specified number of requisite capabilities (in this case, 100). By carefully choosing means and standard deviations for each cluster, we can control the degree of overlap between markets from different clusters. For example, if cluster A has a mean of 500 and a standard deviation of 50 and cluster B has a mean of 550 and a standard deviation of 50, their markets will have a relatively high amount of common requisite capabilities compared to a scenario in which cluster B has a mean of 900 and a standard deviation of 50. See VII-C for technical details on how normal distributions are used to generate discrete sets of capabilities.

Given the varying degree of overlap between market clusters, the goal of each firm in our simulation is to choose when and where to enter and exit markets to maximize their capital. We specify that all firms in a market during a given time step are aware of the market's demand curve, variable costs, and the presence of competitors. Firms use this knowledge to choose production quantities based on a Cournot oligopoly model. Prices are then directly inferred from the demand curve, leaving variation in production quantities and pricing strategies as subjects for future research. Other pertinent economic details, such as the determination of entry, fixed, and variable costs, are included in Section IV-C.

C. Quantitative Variables Calculation

This section describes all of the relevant variables in the simulation and how each is specified or calculated.

- **Minimum and maximum market entry costs**: Specified in the simulator settings.
- **Capabilities per market**: Specified in the simulator settings.
- **Capability Costs**: A minimum and maximum market-level entry cost are specified in the simulator settings. These values are divided by the number of capabilities per market to obtain the corresponding minimum and maximum per-capability costs. Each capability's cost is then drawn independently from a uniform distribution over this range and remains fixed for the duration of the simulation.
- **Entry Costs**: Sum of the costs of the capabilities that the firm must acquire to participate in the market. Firms pay in full for capabilities they do not yet possess. For capabilities the firm does possess, they pay a reduced entry cost according to how *shareable* those capabilities are and how many markets in the firm's portfolio share the capability. See Footnote 1 in Section IV-B for more details.
- **Fixed Costs**: A percentage of the entry cost, as specified in the simulator settings. For example, an entry cost of 200 and a fixed cost percentage of 0.05 results in a fixed cost of 10 per micro time step[2].
- **Exit Costs**: A percentage of the entry cost, as specified in the simulator settings. For example, if a firm paid 200 to enter a market and the exit cost percentage is set to 0.30, they will pay 60 to exit the market.

[2]More precisely, fixed costs are a percentage of the entry costs *the firm would have paid if the market in question were the most recent addition to the firm's portfolio.* This ensures that firms fully benefit from shared capabilities in all instances where markets in their portfolio overlap, regardless of the order in which those markets were entered.

- **Demand slope and intercept**: The simulator settings require a minimum and maximum for the demand slopes and the demand intercepts. For each market, a slope and intercept are chosen from a uniform distribution between these values.
- **Quantity**: Firms choose the quantity they produce in each market in their portfolio at each micro time step according to their production policy, which, for this paper, is set to Cournot production for all firms.
- **Price**: Price in each market at each micro time step is given by $P = a - bQ$, where a and b are the demand curve intercept and slope, respectively, and Q is the total production quantity across all firms in the market.
- **Revenue**: Revenue for each firm-market combination at each micro time step is given by $r = Pq$, where r is the revenue to the given firm in the given market, P is the market price, and q is the quantity produced by the given firm.
- **Profit**: Profit for each firm at each micro time step is the difference between total revenues and total costs across all markets in which the firm participates.

V. Reinforcement Learning for Market Entry

With the MEG simulator in place, we are ready to investigate our principal research question: *Can AI discover effective market entry and exit strategies?* In particular, can existing RL algorithms discover effective strategies in our simulated economy? We first provide a high-level overview of RL in general and the three specific RL algorithms we use in our study. We then outline how state observations are generated in our simulator and how we connect our simulator to off-the-shelf implementations of these algorithms.

A. Reinforcement Learning

RL is a class of AI solution methods that involves learning through interaction with an environment. Such methods are similar to how humans and animals learn: through trial and error. Three characteristics of problems that lend themselves to RL approaches are: 1) the agent's actions influence its later state, 2) the agent is not told which actions to take, and 3) because actions influence future states, they may not affect only immediate rewards but also indirectly affect future rewards [29]. Given such a problem, RL techniques empower agents to map states to actions with the goal of maximizing some reward.

B. RL Algorithms We Employ in this Study

1) Deep-Q Network (DQN): Basic forms of RL often use a simple data structure such as a two-dimensional array to track expected future rewards, known as Q-values, for state-action pairs. Such an approach, however, does not scale well beyond low-dimensional state spaces. Deep-Q Networks (DQNs) address this problem by employing a deep artificial neural network to estimate the Q-values. This approach is advantageous in complex, real-world situations where agents must process high-dimensional inputs and use these to generalize past experience to new situations. See [30] for an in-depth explanation of the DQN architecture.

2) Proximal Policy Optimization (PPO): Proximal Policy Optimization (PPO) is a policy gradient method—a type of RL in which a parameterized policy is optimized with respect to the long-term cumulative reward via gradient descent. PPO differs from prior policy gradient methods, however, in that rather than performing a single step of gradient descent per data sample, PPO collects data through several interactions with the environment before performing minibatch updates on the policy. The algorithm also introduces a clipped objective function that limits the size of policy updates to prevent large, destabilizing swings in policy parameters [31].

3) Advantage Actor-Critic (A2C): Actor-critic methods incorporate separate structures to represent the policy function, which maps states to actions, and the value function, which maps states to values [29]. Advantage Actor-Critic (A2C) is a specific actor-critic model that uses deep neural networks as the policy and value functions.

In their paper introducing the A2C model, Mnih et al. point out that a common issue in RL is non-stationarity in the sequence of data observed by the RL agent. That is, as the agent acts within the environment, its actions modify the underlying probability distribution from which it samples experiences. This can cause strong correlation between the agent's policy updates, destabilizing learning. The A2C model overcomes this issue by having multiple agents interact with multiple instances of the environment in parallel while sharing the same policy and value functions [32].

C. Generation of State Observations

We now detail the state observations that we feed to the RL agent when it is its turn to act. A state observation is a vector that contains pertinent economic data that the agent uses to decide when and where to enter and exit markets. Recall that n is the number of firms in the simulation, k the number of markets, and we will let c indicate the number of capabilities in the economy. State observations then contain:

1) Capital of all firms (vector of dimension n)
2) Market overlap structure (i.e., percentage of overlap in required capabilities between any pair of markets; matrix of dimension $k \times k$)
3) Capability shareability values (vector of dimension c; see Footnote 1 in section IV-B for a detailed explanation of this parameter)
4) Variable costs for all firm-market combinations (matrix of dimension $n \times k$)
5) Fixed cost for each firm-market combination (entries set to zero where the given firm is not present in the given market; matrix of dimension $n \times k$)
6) Market portfolio of all firms (matrix of dimension $n \times k$)
7) Entry cost for every firm-market combination given current market portfolios (matrix of dimension $n \times k$)
8) Demand intercept in each market (vector of dimension k)
9) Slope in each market (vector of dimension k)
10) Most recent quantity for each firm-market combination (matrix of dimension $n \times k$)
11) Most recent price for each firm-market combination (matrix of dimension $n \times k$)

These 11 components of the state representation are each flattened into one-dimensional vectors and then concatenated to create a single state observation vector. Information is ordered according to the following three rules:

1) For components involving firm-specific information, the RL agent's information is given first. Information about other agents is then given in ascending order by agent ID.
2) For components involving market-specific information, information is given in ascending order by market ID.
3) For components involving information specific to firm-market combinations, the above two rules apply. Information is ordered first at the firm level and then at the market level (i.e., info pertaining to a firm for all markets is given before the info for the next firm is given).

The total length of state representation is $k^2 + 6kn + n + 2k + c$. For example, if $n = 5$, $k = 8$, and $c = 2000$, then the resulting state representation is a vector in $\mathbb{R}^{2325}$.

D. Connection to Off-the-shelf RL Implementations

For this study, we use existing implementations [33] from Stable-Baselines3 of the algorithms outlined in Section V-B. Adapting these algorithms or creating new algorithms suited specifically for the task of market diversification presents a promising area for future research; our purpose here is to show that existing methods can handle the problem reasonably well.

VI. Experiments and Results

We now outline the specific experiments we ran in the MEG simulator, using the RL algorithms outlined in the previous section. We first describe the rule-based agents, which provide a baseline level of performance and serve as opponents against which the RL agents can train. We then detail how the RL agents were trained. Finally, we evaluate the performance of the RL agents in the MEG simulator.

A. Baseline Agents

We employ two types of heuristic agents to compete against our RL-based agents. We will refer to these two types of agents as naïve and sophisticated.

When it is a naïve agent's turn to act, it chooses whether to enter a market, exit a market, or do nothing, based on a predefined probability distribution over these three action types. Once an action type is chosen, the agent randomly chooses which market to enter or exit, if applicable, using a uniform probability distribution over the available markets. If there are no markets available (such as when an agent chooses to exit a market when they are not in any markets), they choose the do-nothing action by default.

The sophisticated agents use the same logic as the naïve agents to decide which type of action (i.e., entry, exit, or do-nothing) to take. However, if an entry action is chosen, a sophisticated agent, rather than randomly choosing a market to enter, will enter the market whose requisite capabilities overlap the most with the agent's current set of capabilities. When multiple markets offer the same highest degree of overlap, the agent will randomly choose one of these. When an exit action is chosen, a sophisticated agent, rather than randomly choosing a market to exit, will exit the market in which it had the lowest profit in the most recent micro time step.

We use the naïve agents as a rudimentary baseline to check whether our RL-based agents perform better than random. We then turn to the sophisticated agents to show that our RL-based agents can outperform agents programmed with reasonable, albeit simplified, approximations of real-world firm behavior. The sophisticated agents represent a first-order approximation of how real-world firms make market entry and exit decisions. As described in the introduction, firms tend to choose markets for entry in which they can leverage their capabilities. They decide they want to expand and then choose markets with this simple rule in mind. Likewise, firms tend to exit markets that show low profitability as these weigh down portfolio performance.

B. Training the AI

We train agents in a variety of economic settings to ensure that our approach is generally effective and not reliant on one particular economic setup. In particular, we vary the RL algorithm used, the type and number of opponent agents, the inherent difficulty of the economy, and the degree of overlap between market clusters. For each training configuration, we train 10 agents and evaluate each agent's performance using 1000 evaluation simulations,

where overall performance per trained agent is the mean across all 1000 simulations. The performance of an RL algorithm for a specific economic setup is then reported as the mean across the 10 agents trained using that algorithm and setup. Capital and bankruptcy plots additionally include 95% confidence intervals computed from the distribution of the 10 agent-level means. Tabular results include the standard deviation of these means in parentheses. **Bold green** highlighting within tabular results indicates the best-performing agent type for each economic setup.

C. Evaluation of the AI

We now evaluate the performance of the AI agents relative to their rule-based opponents. We begin by exploring variation in performance by RL algorithm by training agents in the default economy using each of the three algorithms outlined in Section V-B. To evaluate overall performance, we ask two questions: 1) How much capital did each agent accumulate on average? and 2) How often did each type of agent go bankrupt? Note that in all tabular results in this paper, average capital growth refers to the average percentage by which capital increased from the beginning to the end of the simulation. For example, if an agent began a simulation with capital of 1000 and ended with 3000, its capital growth for that simulation would be 200%.

These results below show that A2C performs poorly, DQN offers mixed results, and PPO performs well.

TABLE I
PERFORMANCE BY RL ALGORITHM IN THE DEFAULT ECONOMY (STANDARD DEVIATIONS IN PARENTHESES).

RL Algorithm	Agent Type	Bankruptcy Rate (%)	Avg. Capital Growth (%)
—	Sophisticated	61.77 (0.75)	1895.22 (43.82)
—	Naïve	74.08 (1.04)	1247.02 (80.44)
PPO	AI	**31.07 (2.08)**	**3914.02 (148.12)**
—	Sophisticated	**51.94 (1.44)**	**2787.89 (87.20)**
—	Naïve	65.15 (2.30)	1922.99 (112.69)
A2C	AI	62.21 (7.30)	1082.52 (337.99)
—	Sophisticated	56.05 (1.43)	**2515.10 (110.33)**
—	Naïve	68.77 (1.79)	1698.44 (85.52)
DQN	AI	**37.90 (4.83)**	2219.56 (250.57)

The remainder of the reported results will utilize PPO as the RL algorithm. We first provide visualizations of PPO agent performance in the default economy. We then show agent performance in a variety of economic settings. We show the results for each of these settings within each subsection; see VII-C for a master table containing results across all experiments.

1) Baseline Results: Our baseline results show how the AI agents perform in what we will call the *default economy:* a moderately difficult economy where each AI agent competes against two naïve agents and two sophisticated agents. (For full parameter setting details, see Section VII-C.) We first note that the sophisticated agents consistently outperform their naïve counterparts, suggesting that the simulator correctly rewards agents for intelligently using the market overlap structure and profit metrics to guide decisions rather than entering and exiting at random. Furthermore, we note that the gap between AI performance and sophisticated performance is much greater than the gap between sophisticated and naïve performance, indicating that the strategies the AI agent learned are more effective than the simple rules the sophisticated agents are based on.

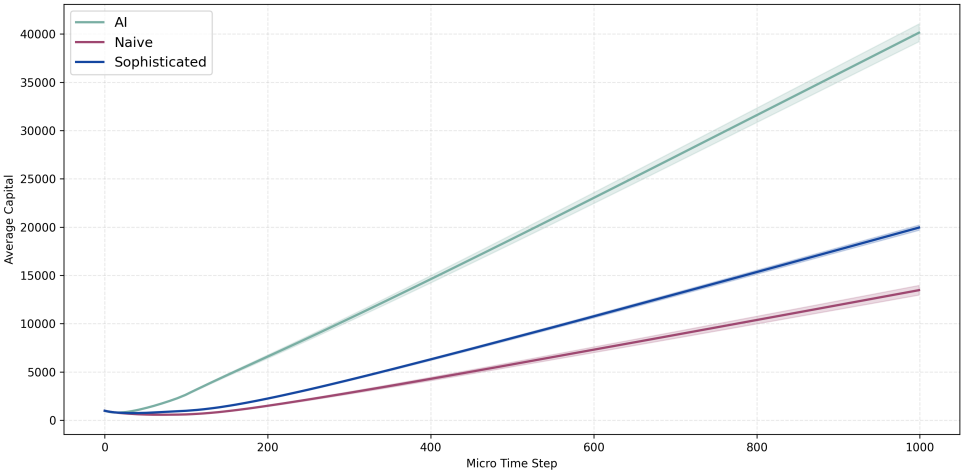

Fig. 1. **Cumulative capital by agent type over time, averaged across 10 trained agents, each evaluated for 1000 simulations. Shading indicates the 95% confidence interval over the distribution of the 10 agent-level means.** The AI agent significantly outperforms both the naïve and sophisticated agents in the default economy.

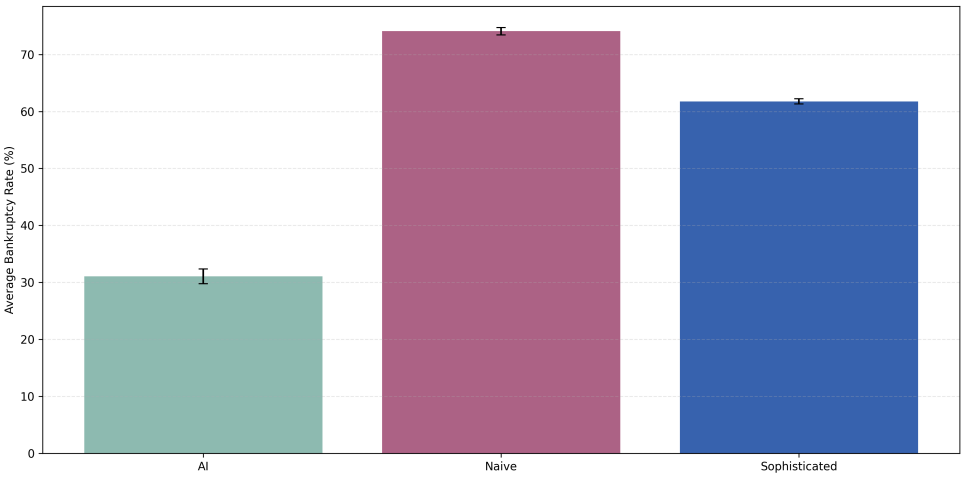

Fig. 2. **Percentage of simulations ending in bankruptcy, averaged across 10 trained agents, each evaluated for 1000 simulations. Whiskers indicate the 95% confidence interval over the distribution of the 10 agent-level means.** The AI agent has a significantly lower bankruptcy rate than the other agent types in the default economy.

2) Variation in Opponent Agent Strategy: We next explore variation in the strategy used by the sophisticated agents. This is done by varying the probability with which they select each type of action on their turn. Table II shows the different sophisticated agents that were used, along with their corresponding probability distributions.

TABLE II
SOPHISTICATED AGENT ACTION-TYPE PROBABILITIES.

Agent	Entry Probability	Exit Probability	Do-nothing Probability
A	0.5	0.0	0.5
B	0.8	0.2	0.0
C	0.1	0.0	0.9
D	0.4	0.3	0.3

Intuitively, these agents can be thought of as modeling different high-level behaviors: agent A is ***a**ggressive* and never exits markets; agent B is a ***b**aseline* (or *default*) agent, entering most of the time but also exiting some of the time; agent C is ***c**onservative*, usually doing nothing and only rarely entering new markets; and agent D is a ***d**o-everything* agent, which splits its behavior nearly equally between the three action types.

We train and evaluate AI agents in an economy containing one of each type of sophisticated agent above. The results of this evaluation are shown in Figures 3 and 4. These results demonstrate that the AI agent has a significantly lower bankruptcy rate and a higher capital growth rate than any of the four sophisticated agents.

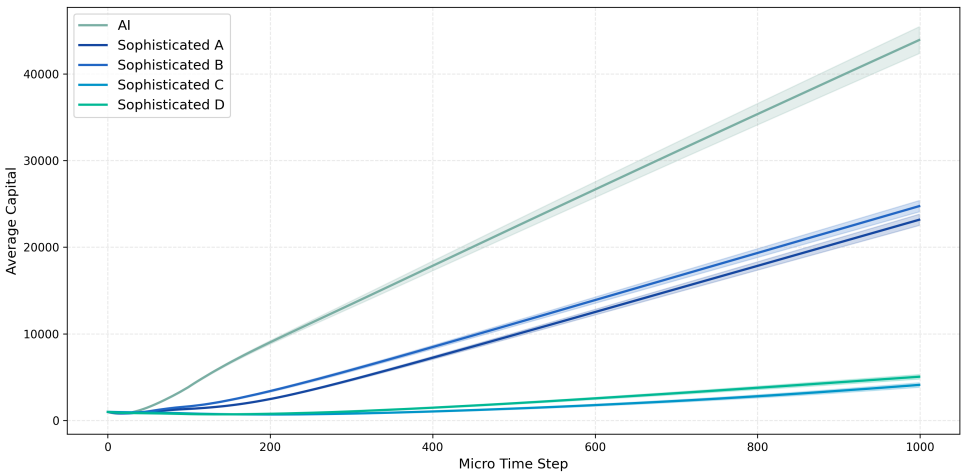

Fig. 3. **Cumulative capital by agent type over time, averaged across 10 trained agents, each evaluated for 1000 simulations. Shading indicates the 95% confidence interval over the distribution of the 10 agent-level means.** The AI agent significantly outperforms the sophisticated agents. The ***a**ggressive* and ***b**aseline* agents perform moderately well while the ***c**onservative* and ***d**o-everything* agents perform poorly.

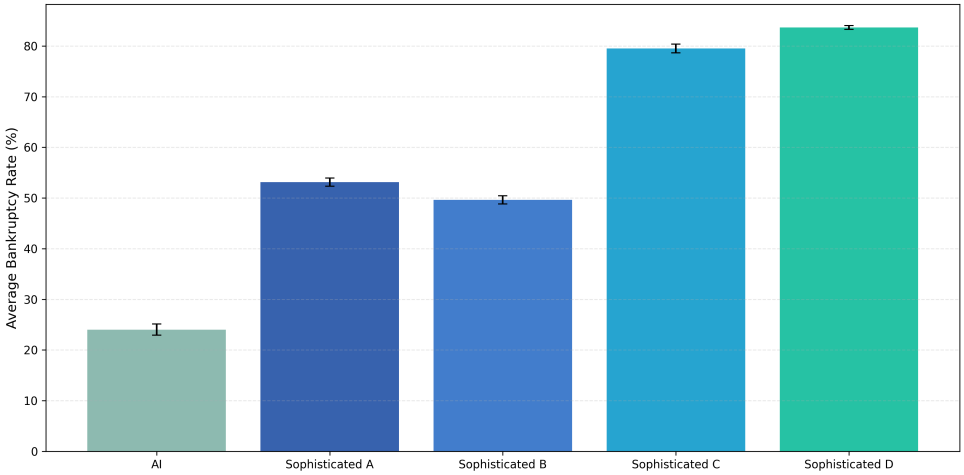

Fig. 4. **Percentage of simulations ending in bankruptcy, averaged across 10 trained agents, each evaluated for 1000 simulations. Whiskers indicate the 95% confidence interval over the distribution of the 10 agent-level means.** The AI agent goes bankrupt in about one-fourth of simulations, significantly outperforming the sophisticated agents. The ***a****ggressive* and ***b****aseline* agents go bankrupt in about half of simulations while the ***c****onservative* and ***d****o-everything* agents go bankrupt in most simulations.

3) Variation in Number of Opponent Agents: Next, we vary the number of opponent agents. Using sophisticated agents of type B, as described in VI-C2, we train and evaluate AI agents in MEG environments featuring one, two, three, four, five, six, and seven opponent agents. All other parameter settings are left at their default values. Table III shows the results. As the number of sophisticated agents in the MEG increase, the economy becomes more difficult for the AI agent, as can be seen in its increasing bankruptcy rate. However, it consistently outperforms the sophisticated agents, achieving more than twice their average capital growth.

4) Variation in Inherent Difficulty of the Economy: We now vary the inherent difficulty of the economy. We do so by training in the *default* economy but altering the starting amount of capital each agent is given as well as the maximum possible demand intercept. Note that the demand intercept affects the prices that firms will receive for selling products, so by altering the upper bound of the distribution from which demand intercepts are drawn when initializing markets, we simulate economies in which selling prices are relatively higher or lower compared to production costs. Table IV lists the four different economy difficulty levels that were used in the experiments, along with their corresponding parameters.

The evaluation results from training AI agents in each of the different difficulty economies are shown in Table V. The bankruptcy rates for each agent increase as the economy difficulty increases, as is to be expected. In terms of capital growth, however, the AI agent interestingly outperforms the other agents by increasingly wider margins as the economy gets more difficult.

5) Variation in Degree of Market Cluster Overlap: The final evaluation variations focus on the degree of overlap between market clusters. (See IV-B for a full explanation of market clusters.) We create economies containing five clusters each with two markets per cluster. The overlap between clusters is varied by changing the probability distributions from which market capabilities are drawn. Each cluster is given the same standard deviation but the

TABLE III
PERFORMANCE AS THE NUMBER OF SOPHISTICATED OPPONENT AGENTS VARIES (STANDARD DEVIATIONS IN PARENTHESES).

# Opponents	Agent Type	Bankruptcy Rate (%)	Avg. Capital Growth (%)
1	Sophisticated	43.83 (1.29)	1563.05 (152.30)
1	**AI**	**10.22 (0.75)**	**4409.78 (196.64)**
2	Sophisticated	48.37 (0.84)	1554.71 (46.59)
2	**AI**	**18.76 (1.12)**	**3595.05 (151.57)**
3	Sophisticated	58.54 (1.03)	1574.79 (57.78)
3	**AI**	**26.66 (3.06)**	**3326.91 (171.91)**
4	Sophisticated	66.19 (0.51)	1611.77 (58.04)
4	**AI**	**34.02 (1.78)**	**3438.73 (150.72)**
5	Sophisticated	71.28 (0.63)	1644.26 (47.56)
5	**AI**	**42.01 (3.38)**	**3566.08 (236.56)**
6	Sophisticated	75.01 (0.28)	1657.59 (24.25)
6	**AI**	**46.09 (1.80)**	**3705.98 (102.15)**
7	Sophisticated	77.83 (0.30)	1684.59 (40.01)
7	**AI**	**50.77 (2.07)**	**3847.72 (198.86)**

TABLE IV
ECONOMY DIFFICULTY CLASSES.

Difficulty	Starting Capital	Demand Intercept Min	Demand Intercept Max
Very Easy	2000	10	20
Easy	1300	10	18
Default	1000	10	15
Difficult	700	10	12

TABLE V
PERFORMANCE UNDER VARYING ECONOMY DIFFICULTY (STANDARD DEVIATIONS IN PARENTHESES).

Economy Difficulty	Agent Type	Bankruptcy Rate (%)	Avg. Capital Growth (%)
Very Easy	Sophisticated	41.48 (1.00)	1253.86 (37.16)
Very Easy	Naïve	53.30 (0.52)	921.28 (24.62)
Very Easy	**AI**	**22.00 (2.22)**	**1896.85 (88.72)**
Easy	Sophisticated	49.76 (1.03)	1821.62 (36.91)
Easy	Naïve	61.99 (1.31)	1294.58 (37.17)
Easy	**AI**	**25.87 (2.21)**	**3039.21 (120.44)**
Default	Sophisticated	61.77 (0.75)	1895.22 (43.82)
Default	Naïve	74.08 (1.04)	1247.02 (80.44)
Default	**AI**	**31.07 (2.08)**	**3914.02 (148.12)**
Difficult	Sophisticated	75.89 (0.96)	2045.41 (116.25)
Difficult	Naïve	85.38 (0.77)	1245.13 (84.79)
Difficult	**AI**	**47.62 (2.20)**	**6354.85 (287.10)**

cluster means are adjusted to move the clusters farther apart (to achieve low overlap) or closer together (to achieve high overlap). There are 2000 total capabilities in the economy and we select 100 of these for each market. Table VI shows the means of the probability distributions from which market capabilities are drawn, where μ_i denotes the mean of the underlying probability distribution for cluster i.

TABLE VI
CLUSTER MEANS USED TO CONTROL THE DEGREE OF MARKET OVERLAP (μ_i IS THE MEAN FOR CLUSTER i).

Degree of Cluster Overlap	μ_1	μ_2	μ_3	μ_4	μ_5
Low	200	600	1000	1400	1800
Default	800	900	1000	1100	1200
High	900	950	1000	1050	1100

We trained and evaluated AI agents in economies with each of the different degrees of market overlap. The results of this evaluation are shown in Table VII. Regardless of the amount of overlap, the AI agents exhibit superior performance, both in terms of reduced bankruptcy rate compared to the baseline agents as well as an increased average capital growth.

TABLE VII
PERFORMANCE UNDER VARYING DEGREES OF MARKET CLUSTER OVERLAP (STANDARD DEVIATIONS IN PARENTHESES).

Market Overlap	**Agent Type**	**Bankruptcy Rate (%)**	**Avg. Capital Growth (%)**
Low	Sophisticated	62.82 (0.75)	1764.36 (75.35)
Low	Naïve	76.12 (0.88)	1088.93 (56.60)
Low	**AI**	**36.14 (2.13)**	**3693.81 (241.05)**
Default	Sophisticated	61.77 (0.75)	1895.22 (43.82)
Default	Naïve	74.08 (1.04)	1247.02 (80.44)
Default	**AI**	**31.07 (2.08)**	**3914.02 (148.12)**
High	Sophisticated	58.93 (1.20)	1945.43 (65.21)
High	Naïve	70.05 (1.05)	1361.28 (52.93)
High	**AI**	**29.85 (1.38)**	**3865.66 (116.14)**

D. Discussion and Validation

The experimental results presented in Section VI-C demonstrate that the AI agents outperform the baseline agents in a wide variety of economic settings. This robust ability to obtain high-level performance, regardless of the specific economic configuration, provides a positive answer to our main motivating research question: *Can AI discover effective market entry and exit strategies?* Yes, RL algorithms can discover successful strategies in the MEG simulator.

To ensure that these results are due to reasonable behavior and not due to exploiting some unintended loophole in the simulator, we now provide a high-level validation of the AI agent's strategic action choices. Figure 5 shows a visualization of the average AI behavior across all 10,000 simulations in the default economy (10 independently trained AI agents, each evaluated for 1000 simulations). The 10 markets are organized into five clusters of two markets each. We see that the AI agent enters all markets aggressively, but the sequence of market entrances is difficult to determine from these aggregated data.

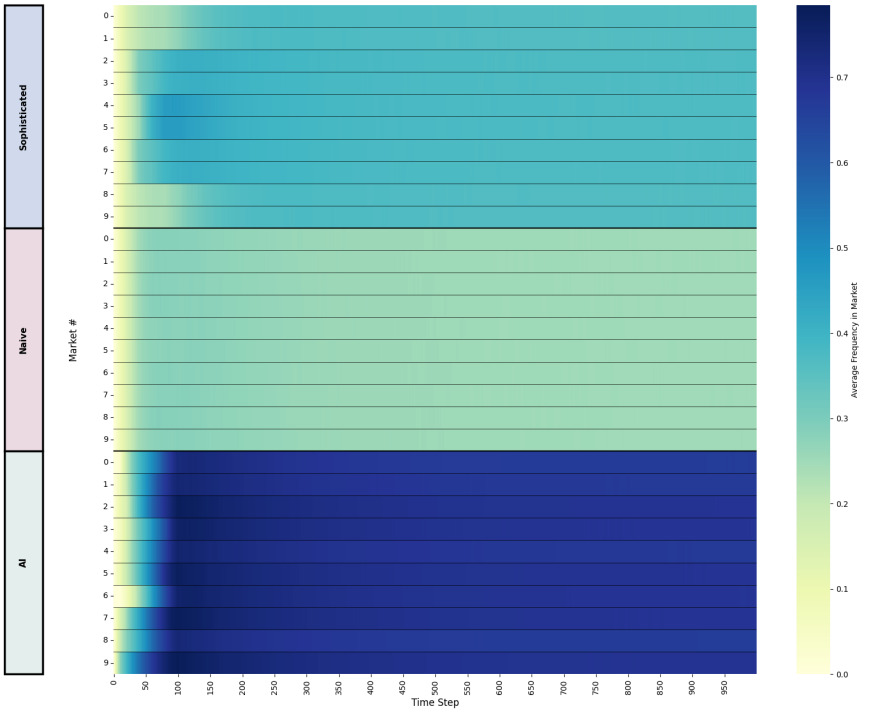

Fig. 5. **A visualization of market entry behavior.** Each row corresponds to a combination of agent type and market. The x-axis shows micro time steps. The color gradient indicates the percentage of simulations in which a given agent was present in a given market at a given time step. There are 10 markets in this economy, organized into five clusters of two markets each. We observe that the AI agent enters all markets aggressively.

To better interpret the strategy of the AI agent, it is useful to examine individual runs. We commonly observe a pattern similar to the one in Figures 6 and 7, where the AI agent avoids markets where its competitors are present while also entering markets that overlap with its current set of capabilities. This behavior indicates that the AI agent follows a strategy that incorporates the market overlap mechanics of the MEG simulator. The AI agent's strategy often leads to success, as shown in Figure 6. In that simulation, the AI agent dominates all markets while the other agents go bankrupt. However, the strategy does sometimes fail, as shown in Figure 7, where the AI agent goes bankrupt. More detailed investigation of the properties of these strategies is a promising area for further research, particularly in the field of strategic management.

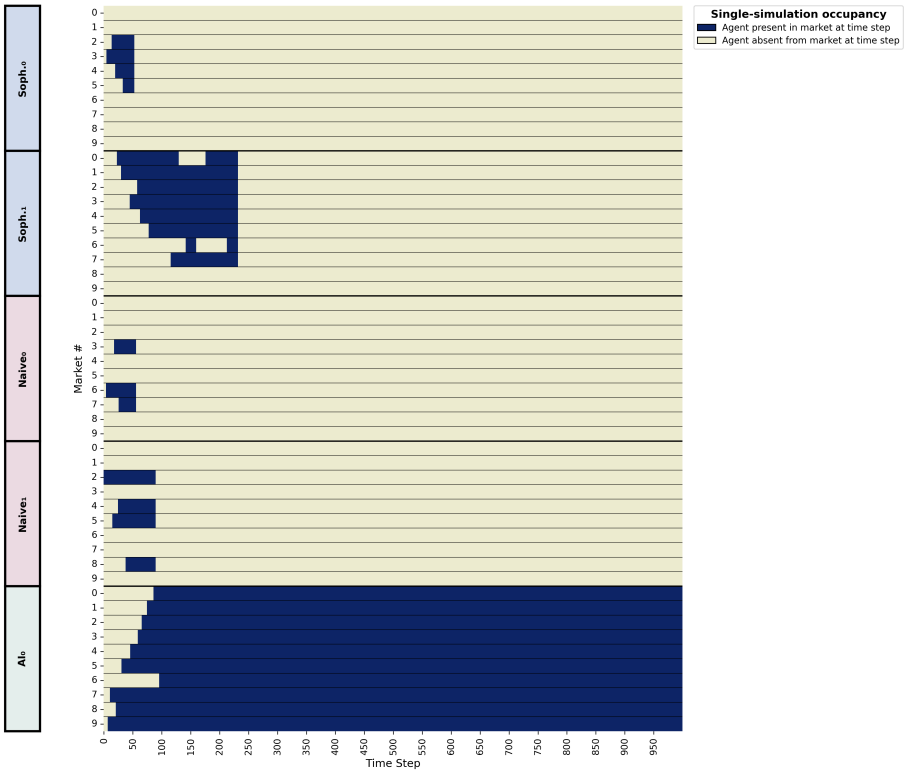

Fig. 6. **A visualization of market entry behavior from a successful simulation.** The AI agent tends to enter where its competitors are absent while also preferring markets that overlap with its current portfolio. Over the course of this simulation, the AI agent increased its capital by a factor of over 131, while all other agents went bankrupt.

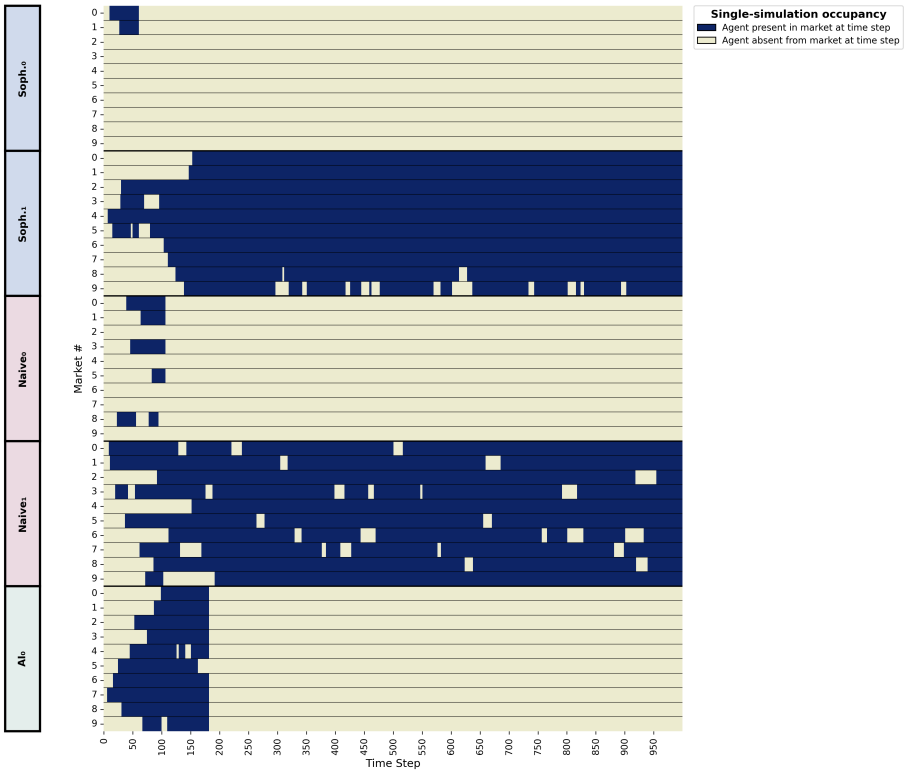

Fig. 7. **A visualization of market entry behavior from an unsuccessful simulation.** The AI agent again tends to enter markets where its competitors are absent and that overlap with its current portfolio. However, perhaps due to not fully considering other pertinent details such as demand and cost curves in each market, this strategy leads to the AI agent's bankruptcy.

VII. Conclusion and Future Work

In this work, we provided a proof of concept that modern AI can discover effective market entry and exit strategies. We showed that off-the-shelf deep RL algorithms are robust learning tools for market entry strategy across a variety of economic conditions.

While the current version of our economic simulator cannot yet provide actionable insights for real-world business scenarios, it lays a foundational groundwork for achieving that goal. We envision a future in which business leaders and economists heavily incorporate AI tools into their decision-making, leading to increased productivity and efficient capital allocation in dynamic, global economies. To that end, future efforts will focus on economic realism, scalability, and more complex game-theoretic dynamics.

A. Economic Realism

Future work will integrate real-world economic data into our simulations. For example, rather than modeling capabilities abstractly, we could incorporate actual data on the costs of machinery, labor, materials, management, and R&D in various industries. We could also leverage publicly available financial information to model the scenarios faced by real businesses. Additionally, we can make our simulations more realistic by introducing other relevant economic factors, including debt and equity financing, government monetary and fiscal policy, and the business cycle.

B. Scalability

A significant limitation of this work is the need to restart AI training from scratch each time the simulator settings are changed. This is because the dimensions of the state observations we supply the RL agents are dependent on the simulator settings. In the future, we plan to create a generalized state representation such that the same model can be trained under a variety of conditions (and hopefully learn more generalizable strategies). This will be a crucial step toward making our approach scale to scenarios with more agents, markets, and economic factors. Without these adjustments, the need for frequent retraining could quickly become prohibitive as the simulator becomes more complex.

C. Game-theoretic Dynamics

In this study, we assume that the AI agent is learning against stationary-strategy agents. Therefore, the AI agent may learn strategies that could quickly become ineffective in real scenarios where other agents can adapt their behaviors. In future work, we aim to explore adaptive strategies using multi-agent RL (MARL). We are also interested in exploring how different reward functions influence learned behavior. For instance, what emergent behaviors might we observe if agents are awarded according to the *difference* between their capital and that of their opponents rather than according to just their own capital?

Responsible AI Usage Acknowledgment

We used generative AI tools to assist with the following tasks:

- Fixing bugs, enhancing our code base, and making cosmetic changes to plot-generating code.
- Suggesting minor edits to the prose of this paper.
- Orchestrating training runs and data collection.

For most tasks, we used the latest version of GPT-Codex from OpenAI (most recently, GPT-5.5). Additionally, we used the latest version of ChatGPT for some tasks. These tools were used to both expedite research and reduce the probability of human errors affecting the quality of our research. We have ensured the accuracy of AI-generated code and text in this project and assume full responsibility for all code, text, data, and plots related to this work.

Source Code and Data Availability Statement

The source code and experimental data are publicly available. These materials include simulator configurations, training hyperparameters, trained agent artifacts, and evaluation datasets for all experiments. The source code is licensed under the MIT License, and the dataset is licensed under the Creative Commons Attribution 4.0 International License (CC BY 4.0). The dataset is archived with DOI 10.57967/hf/8605.

Author contributions

Eric Thomas: Conceptualization, software, methodology, formal analysis, investigation, visualization, and writing–original draft. Christopher Archibald: Conceptualization, methodology, supervision, and writing–review and editing. Stephen Sorensen: Conceptualization, domain expertise, and writing–review and editing. David Bryce: Conceptualization, supervision, domain expertise, and writing–review and editing.

Acknowledgments

The authors thank Blake Peterson, Jacob DeGraw, and Austin McMaster, who were undergraduate research assistants at Brigham Young University at the time of their contributions. Blake Peterson and Jacob DeGraw assisted with plot-generation code, and Austin McMaster assisted with simulator configuration management and high-performance computing setup. These contributions supported the implementation and execution of the experiments reported in this paper.

Funding

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Disclosure statement

Stephen Sorensen was employed by Bain & Company during part of this project. Eric Thomas is employed by Cúratus. Bain & Company and Cúratus had no role in the study design, analysis, writing, or decision to submit this article. The authors report no competing interests related to this work.

Notes on contributors

Eric Thomas is a graduate student in the Department of Computer Science at Brigham Young University. His research interests include reinforcement learning, robotics, multi-agent systems, and skill estimation in physical domains.

Christopher Archibald is an Associate Professor in the Computer Science Department at Brigham Young University. His research focuses on multi-agent systems, game theory, and strategic decision-making, with applications in games and sports.

Stephen Sorensen is a strategy consultant with experience at Bain & Company, where he has worked on M&A transactions, AI-driven process improvement, and private equity engagements. He also serves as an adjunct professor in the Marriott School of Business at Brigham Young University where he has co-developed and taught coursework in advanced analytics and business applications of artificial intelligence.

David Bryce is an associate professor of strategic management in the Marriott School of Business at Brigham Young University. His research interests include artificial intelligence in strategy, platform ecosystems, corporate strategy, innovation, and technological change. His work focuses on how firms use capabilities, governance, and sequential strategic investments to create value and adapt to changing competitive environments.

REFERENCES

[1] K. M. Eisenhardt and D. N. Sull, *Strategy as simple rules*. Harvard Bus Pub, 2001, vol. 6, no. 4.

[2] R. P. Rumelt, "Strategy, and structure and economic perfor mance (boston, division of research, graduate school of business administration, harvard university)," *Structure, and Economic Performance1974*, 1974.

[3] J. C. Panzar and R. D. Willig, "Economies of scope," *The American economic review*, vol. 71, no. 2, pp. 268–272, 1981.

[4] D. J. Teece, "Towards an economic theory of the multiproduct firm," *Journal of economic behavior & organization*, vol. 3, no. 1, pp. 39–63, 1982.

[5] L. Tesfatsion, "Chapter 16 agent-based computational economics: A constructive approach to economic theory," in *Handbook of Computational Economics*, ser. Handbook of Computational Economics, L. Tesfatsion and K. Judd, Eds. Elsevier, 2006, vol. 2, pp. 831–880. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1574002105020162

[6] S. Zheng, A. Trott, S. Srinivasa, N. Naik, M. Gruesbeck, D. C. Parkes, and R. Socher, "The ai economist: Improving equality and productivity with ai-driven tax policies," *arXiv preprint arXiv:2004.13332*, 2020.

[7] M. B. Johanson, E. Hughes, F. Timbers, and J. Z. Leibo, "Emergent bartering behaviour in multi-agent reinforcement learning," *arXiv preprint arXiv:2205.06760*, 2022.

[8] Goldman Sachs, "Artificial intelligence (ai) market interest growth 2015 to 2023, by share of companies [graph]," August 2023, in Statista. [Online]. Available: https://www.statista.com/statistics/1424672/ai-market-interest-worldwide/

[9] L. Chen, A. Mislove, and C. Wilson, "An empirical analysis of algorithmic pricing on amazon marketplace," in *Proceedings of the 25th international conference on World Wide Web*, 2016, pp. 1339–1349.

[10] E. W. Ngai, L. Xiu, and D. C. Chau, "Application of data mining techniques in customer relationship management: A literature review and classification," *Expert systems with applications*, vol. 36, no. 2, pp. 2592–2602, 2009.

[11] E. W. Ngai, Y. Hu, Y. H. Wong, Y. Chen, and X. Sun, "The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature," *Decision support systems*, vol. 50, no. 3, pp. 559–569, 2011.

[12] IBM, "Data suggests growth in enterprise adoption of ai is due to widespread deployment by early adopters, but barriers keep 40% in the exploration and experimentation phases," jan 2024, iBM Newsroom. [Online]. Available: https://tinyurl.com/39wa2f63

[13] J. S. Roppelt, N. S. Greimel, D. K. Kanbach, S. Stubner, and T. K. Maran, "Artificial intelligence in talent acquisition: a multiple case study on multi-national corporations," *Management decision*, 2024.

[14] G. Cohen, "Algorithmic trading and financial forecasting using advanced artificial intelligence methodologies," *Mathematics*, vol. 10, no. 18, p. 3302, 2022.

[15] K. Olorunnimbe and H. Viktor, "Deep learning in the stock market—a systematic survey of practice, backtesting, and applications," *Artificial Intelligence Review*, vol. 56, no. 3, pp. 2057–2109, 2023.

[16] G. N. Yannakakis and J. Togelius, *Artificial intelligence and games*. Springer, 2018, vol. 2.

[17] M. Campbell, A. J. Hoane Jr, and F.-h. Hsu, "Deep blue," *Artificial intelligence*, vol. 134, no. 1-2, pp. 57–83, 2002.

[18] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[19] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel *et al.*, "Mastering chess and shogi by self-play with a general reinforcement learning algorithm," *arXiv preprint arXiv:1712.01815*, 2017.

[20] O. Vinyals, T. Ewalds, S. Bartunov, P. Georgiev, A. S. Vezhnevets, M. Yeo, A. Makhzani, H. Küttler, J. Agapiou, J. Schrittwieser *et al.*, "Starcraft ii: A new challenge for reinforcement learning," *arXiv preprint arXiv:1708.04782*, 2017.

[21] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev *et al.*, "Grandmaster level in starcraft ii using multi-agent reinforcement learning," *nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[22] M. Moravčík, M. Schmid, N. Burch, V. Lisỳ, D. Morrill, N. Bard, T. Davis, K. Waugh, M. Johanson, and M. Bowling, "Deepstack: Expert-level artificial intelligence in heads-up no-limit poker," *Science*, vol. 356, no. 6337, pp. 508–513, 2017.

[23] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Dębiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse *et al.*, "Dota 2 with large scale deep reinforcement learning," *arXiv preprint arXiv:1912.06680*, 2019.

[24] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[25] R. S. Sutton, "Introduction: The challenge of reinforcement learning," in *Reinforcement learning*. Springer, 1992, pp. 1–3.

[26] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.

[27] Y. Ye, H. Pei, B. Wang, P.-Y. Chen, Y. Zhu, J. Xiao, and B. Li, "Reinforcement-learning based portfolio management with augmented asset movement prediction states," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 01, 2020, pp. 1112–1119.

[28] D. J. Bryce and S. G. Winter, "A general interindustry relatedness index," *Management Science*, vol. 55, no. 9, pp. 1570–1585, 2009.

[29] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[30] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[31] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[32] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*. PMLR, 2016, pp. 1928–1937.

[33] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," *Journal of Machine Learning Research*, vol. 22, no. 268, pp. 1–8, 2021.

APPENDIX A: MARKET CAPABILITY INITIALIZATION ALGORITHM

Algorithm 1 Select Market Capabilities

Require: $\mu \in \mathbb{R}$, $\sigma \in \mathbb{R}$ ▷ Input: mean and standard deviation
Ensure: $\mathbf{v} \in \mathbb{R}^n$ ▷ Output: 1-hot encoded vector in $\mathbb{R}^n$
$\mathbf{v} \leftarrow \mathbf{0}^n$ ▷ Initialize $\mathbf{v}$ to the zero vector
for $i \leftarrow 0$ to $n-1$ **do**
    Draw $x \sim \mathcal{N}(\mu, \sigma^2)$
    $x' = \text{round}(x)$
    $roundedUp \leftarrow$ **False**
    if $x' > x$ **then**
        $roundedUp \leftarrow$ **True**
    end if
    $done \leftarrow$ **False**
    $attempts \leftarrow 0$
    while not $done$ **do**
        $attempts \leftarrow attempts + 1$
        if $0 \leq x' < n$ **and** $\mathbf{v}[x'] == 0$ **then**
            $\mathbf{v}[x'] \leftarrow 1$
            $done \leftarrow$ **true**
        else
            if $roundedUp$ **then**
                $roundedUp \leftarrow$ **false**
                $x' \leftarrow x' - attempts$
            else
                $roundedUp \leftarrow$ **true**
                $x' \leftarrow x' + attempts$
            end if
        end if
    end while
end for

APPENDIX B: DEFAULT SIMULATOR SETTINGS

- **RL Algorithm Used**: PPO (Stable-Baselines3 Implementation)
- **Simulation Settings**:
  - Macro time steps per simulation: 100
  - Skip turns per regular turn: 1
  - Starting capital per firm: 1000
  - Fixed cost for existence: True[3]
- **Randomization**:
  - Randomize turn order within each macro time step: True
  - Randomize variable costs per simulation: True
  - Randomize capability costs per simulation: True
  - Randomize markets per simulation: True[4]
- **Sophisticated agent settings**:
  - Entry policy: Highest overlap
  - Exit policy: Loss
  - Entry, exit, and do-nothing action probabilities: [80, 20, 0]
- **Naïve agent settings**:
  - Entry policy: All
  - Exit policy: All
  - Entry, exit, and do-nothing action probabilities: [80, 20, 0]
- **Economy parameters**:
  - Possible capabilities: 2000
  - Capabilities per market: 100
  - Capability shareability: 0.5
  - Number of markets: 10
  - Number of market clusters: 5
  - Markets per cluster: [2, 2, 2, 2, 2]
  - Cluster means: [800, 900, 1000, 1100, 1200]
  - Cluster standard deviations: [50, 50, 50, 50, 50]
- **Market parameters**:
  - Maximum variable cost: 5.0
  - Minimum variable cost: 0.5
  - Maximum entry cost: 500
  - Minimum entry cost: 50
  - Fixed cost as percentage of entry cost: 5
  - Exit cost as percentage of entry cost: 30
  - Maximum demand intercept: 15
  - Minimum demand intercept: 10
  - Maximum demand slope: 0.9
  - Minimum demand slope: 0.6

[3]Deducts a small amount of capital from each agent at each time step such that an agent that does nothing has its capital linearly depleted to zero over the course of a simulation, thus incentivizing agents to act rather than do nothing.

[4]Redraws demand intercepts, demand slopes, and capability vectors for each market from the same underlying probability distributions.

APPENDIX C: MASTER TABLE OF RESULTS

TABLE VIII
COMPARISON OF AGENT PERFORMANCE ACROSS DIFFERENT ECONOMY SETTINGS. NOTE: ALL AI AGENTS WERE TRAINED USING PPO EXCEPT WHERE STATED OTHERWISE.

Economy Setting	**Agent Type**	**Bankruptcy Rate (%)**	**Avg. Capital Growth (%)**
Default	Sophisticated	61.77 (0.75)	1895.22 (43.82)
Default	Naïve	74.08 (1.04)	1247.02 (80.44)
Default	AI	31.07 (2.08)	3914.02 (148.12)
Default	Sophisticated	51.94 (1.44)	2787.89 (87.20)
Default	Naïve	65.15 (2.30)	1922.99 (112.69)
Default (using A2C)	AI	62.21 (7.30)	1082.52 (337.99)
Default	Sophisticated	56.05 (1.43)	2515.10 (110.33)
Default	Naïve	68.77 (1.79)	1698.44 (85.52)
Default (using DQN)	AI	37.90 (4.83)	2219.56 (250.57)
Varying Opponent Strategy	AI	24.03 (1.77)	4291.64 (249.17)
Varying Opponent Strategy	Sophisticated A	53.13 (1.30)	2217.08 (104.53)
Varying Opponent Strategy	Sophisticated B	49.65 (1.30)	2374.23 (105.11)
Varying Opponent Strategy	Sophisticated C	79.52 (1.42)	310.43 (42.93)
Varying Opponent Strategy	Sophisticated D	83.66 (0.61)	404.68 (42.17)
1 Sophisticated Opponent	Sophisticated	43.83 (1.29)	1563.05 (152.30)
1 Sophisticated Opponent	AI	10.22 (0.75)	4409.78 (196.64)
2 Sophisticated Opponents	Sophisticated	48.37 (0.84)	1554.71 (46.59)
2 Sophisticated Opponents	AI	18.76 (1.12)	3595.05 (151.57)
3 Sophisticated Opponents	Sophisticated	58.54 (1.03)	1574.79 (57.78)
3 Sophisticated Opponents	AI	26.66 (3.06)	3326.91 (171.91)
4 Sophisticated Opponents	Sophisticated	66.19 (0.51)	1611.77 (58.04)
4 Sophisticated Opponents	AI	34.02 (1.78)	3438.73 (150.72)
5 Sophisticated Opponents	Sophisticated	71.28 (0.63)	1644.26 (47.56)
5 Sophisticated Opponents	AI	42.01 (3.38)	3566.08 (236.56)
6 Sophisticated Opponents	Sophisticated	75.01 (0.28)	1657.59 (24.25)
6 Sophisticated Opponents	AI	46.09 (1.80)	3705.98 (102.15)
7 Sophisticated Opponents	Sophisticated	77.83 (0.30)	1684.59 (40.01)
7 Sophisticated Opponents	AI	50.77 (2.07)	3847.72 (198.86)
Very Easy Economy	Sophisticated	41.48 (1.00)	1253.86 (37.16)
Very Easy Economy	Naïve	53.30 (0.52)	921.28 (24.62)
Very Easy Economy	AI	22.00 (2.22)	1896.85 (88.72)
Easy Economy	Sophisticated	49.76 (1.03)	1821.62 (36.91)
Easy Economy	Naïve	61.99 (1.31)	1294.58 (37.17)
Easy Economy	AI	25.87 (2.21)	3039.21 (120.44)
Difficult Economy	Sophisticated	75.89 (0.96)	2045.41 (116.25)
Difficult Economy	Naïve	85.38 (0.77)	1245.13 (84.79)
Difficult Economy	AI	47.62 (2.20)	6354.85 (287.10)
Low Market Overlap	Sophisticated	62.82 (0.75)	1764.36 (75.35)
Low Market Overlap	Naïve	76.12 (0.88)	1088.93 (56.60)
Low Market Overlap	AI	36.14 (2.13)	3693.81 (241.05)
High Market Overlap	Sophisticated	58.93 (1.20)	1945.43 (65.21)
High Market Overlap	Naïve	70.05 (1.05)	1361.28 (52.93)
High Market Overlap	AI	29.85 (1.38)	3865.66 (116.14)