Poisoning the Well: Can We Simultaneously Attack a Group of Learning Agents?

Ridhima Bector
Nanyang Technological University
Singapore, 639798
ridhima001@e.ntu.edu.sg

Hang Xu
Nanyang Technological University
Singapore, 639798
hang017@e.ntu.edu.sg

Abhay Aradhya
Nanyang Technological University
Singapore, 639798
abhayaradhya@ntu.edu.sg

Chai Quek
Nanyang Technological University
Singapore, 639798
ashcquek@ntu.edu.sg

Zinovi Rabinovich
Nanyang Technological University
Singapore, 639798
zinovi@ntu.edu.sg

ABSTRACT

As Reinforcement Learning (RL) solutions are becoming ubiquitous, so is the study of potential threats to their training and deployment. While single-learner training-time attacks, capable of "pre-programming" behavioral triggers into a strategy, receive increasing attention, attacks on collections of learning agents have been largely overlooked. We remedy the situation by developing a constructive training-time attack on a population of learning agents and make the attack agnostic to the size of the population. The attack constitutes a sequence of environment (re)parameterizations (poisonings), generated to overcome individual differences between agents and lead the entire population to the same target behavior while minimizing effective environment modulation. Our method is demonstrated on populations of independent learners in "ghost" environments (learners do not interact or perceive each other) as well as environments with mutual awareness, with or without individual learning. From the attack perspective, we pursue an ultra-blackbox setting, i.e., cross-policy traces of the victim learners are the only input both for attack conditioning *and* attack evaluation during the attacker's training. To manage the resulting uncertainty in population behavior, we deploy a novel Wasserstein distance-based Gaussian embedding of detected behaviors within the population of victim learners. To align with prior works on environment poisoning, our experiments are based on a 3D Grid World domain and show: a) feasibility, i.e., despite the uncertainty, the attack forces a population-wide adoption of target behavior; b) efficacy, i.e., the attack is size-agnostic and transferable.

KEYWORDS

Training-Time Attack, Collective Environment Poisoning, Population Attack, Multi-Agent Reinforcement Learning

1 INTRODUCTION

Reinforcement learning (RL) has proliferated most AI applications that investigate unexplored spaces and has bestowed these applications with remarkable capabilities, superhuman at times [10, 16, 17]. Alas, there is no Superman without Kryptonite. RL methods are subject to a variety of attacks that can degrade a policy's performance during deployment; introduce behavior triggers into it or force an agent to learn an a priori non-optimal target strategy [4, 8]. To achieve this, an adversarial system is constructed that encompasses an RL agent, its environment, and its task. In these systems, the RL agent is regarded as the victim, its RL environment, the victim environment, and the task, the victim task. In addition to the victim, the system includes an adversary, tasked with attacking the victim agent. Since the attacker's task is no easier than the victim's, machine learning solutions (and RL, in particular) have been deployed on the attacker's side as well. All attack solutions are commonly classified by 3 features: the form of attack (Train vs Test), the mode of attack (Reward vs Observation vs Environment), and the level of access (Whitebox vs Blackbox) to the victim's inner workings granted to the attacker.

In this paper, we focus on training-time, environment-poisoning attacks. That is we seek to influence the training/optimization of the victim agent policy by means of altering the environment dynamics (the way it changes in response to the victim's actions) akin to [2, 22, 27]. The goal of the attack being to introduce "backdoors" or behavioral triggers into the victim's learned strategy by means encapsulated within the environment mechanics, avoiding access to the victim's inner workings. Furthermore, following this line of reasoning, we favor a blackbox setting, wherein an attacker estimates the victim's behavior policy by observing the victim's interaction with the environment and then conditions the attack on this estimated behavior. In fact, we expand this notion. Normally, during the attacker's training, the system would have access to a proxy victim's inner workings. Such a proxy victim's actual strategy would then be used to generate an *extrinsic* reward signal for the attacker. In contrast, this paper adopts an Ultra-Blackbox (UB) scenario, where the reward signal is *intrinsic*, i.e., generated based on the observed and perceived behavior of the (proxy) victim without any access to its inner workings.

Although there is some progress in attacking Multi-agent RL (MARL) systems (e.g., [3, 5, 11, 12, 19]), to the best of the authors' knowledge, none have yet studied the question of multiple RL agents being attacked simultaneously with an environment poisoning attack. Eyeing social and collective learning settings (e.g., [7, 13, 14, 18, 29]), we seek to attack a *population* of learners without having the luxury of access to any single individual inner workings or the inter-agent relationship. To begin, we adopt scenarios with simplified collectives: a) an *Implicit Collective*, where agents are unaware of each other (essentially inhabit copies of the same environment), and practice individual learning; b) a *Collective*, where agents are aware

Proc. of the Adaptive and Learning Agents Workshop (ALA 2023), Cruz, Hayes, Wang, Yates (eds.), May 29-30, 2023, London, UK, https://alaworkshop2023.github.io/. 2023.

of each other's existence, and learn via individual as well as social learning; c) a *Swarm Collective*, where agents are aware of each other's existence, but are anonymous to each other, and practice social learning. We describe these in greater detail in Section 2

Now, to finalize our approach and define an optimal sequence of environment poisonings, we would need the capability to efficiently capture the distribution of policies used by the victim collective and measure the effect a poisoned environment has on such a distribution. Technically, our approach to these issues is *structurally* more related to the Optimal Transport Kernel Embedding (OTKE) [15], than any other set representation method, such as [20, 23, 30]. Specifically, we use a two-step representation for the set of all policies found in a population: a) representing uncertainty in each agent's policy, given a cross-policy interaction trace; b) capturing the distribution of behaviors at the population level. The former is achieved by an application of VAE, and the latter by a Wasserstein barycenter in the resulting latent space. Notice that we somewhat presume that environment poisoning is effective and only use a single barycenter, disregarding a potential sub-structure at this research stage. In particular, we address the following two hypotheses: **H1)** Wasserstein distance-based Gaussian embedding is capable of capturing the behavior of different-sized victim populations; and **H2)** Attack strategy learned on a given population is transferable to other populations of different sizes.

To sum up, we introduce: a) **Collective Environment Poisoning (CEP)** framework, which we experimentally instantiate for three scenarios: *implicit collectives*, *collectives* and *swarm collectives*; b) **Size-Agnostic Population Behavior Representation** based on a Wasserstein distance-based Gaussian embedding; c) **Ultra-Blackbox (UB) Adversarial Setting**, wherein across-policy behavior traces of the victim population are used to both condition and evaluate the attack.

2 VICTIM POPULATION SETTINGS

Collective learning frameworks where agents train on copies of the same environment fall under three lines of research [7, 13, 14, 18, 29]. The first line aims to decrease the complexity of solving large Markov decision processes that model sizeable multi-agent systems [14]. They do so by breaking down the problem into tasks that are executed in parallel. Each task houses one or more agents and coordinates at run-time in order to push the agent population towards optimal behavior. The second line of research enables a single agent to better explore the given environment in a more efficient manner, by interacting with it in parallel using different policies [7]. In the third line of research, the agent aims to learn how to learn and strives to be able to perform efficiently across a family of tasks (Meta RL) [18]. Herein, the agent trains on different tasks, in parallel. In all three domains, the agents exchange information in order to learn better strategies more quickly. [28] points out that an optimal balance between individual and social learning is critical for learning optimal behaviors in a population. Herein, individual learning is when each agent explores, interacts, and learns from the environment separately while social learning is when agents copy each other's behaviors. Therefore, individual learning leads to innovations which can then efficiently spread through the population via social learning. However, individual learning incurs high exploration cost. On the other hand, social learning is cost-effective but requires construction of a strategy by which the population chooses which behaviors (successful vs majority) to copy. The optimality of these strategies, akin to meta-control strategies in the human brain [6], are sensitive to environment variability. In environments that change frequently, social learning might hamper the population's learning as the exchanged information can become invalid very quickly. This work proposes three learning settings namely; Implicit Collective, Collective, and Swarm Collective; wherein individual learning decreases while social learning increases progressively. In these settings, each agent trains to learn its individual task with the support of a separate reward signal while the attacker trains to push the complete (victim) populations towards a target behavior.

In the Implicit Collective setting, the innovation capability of the population is maximized via individual learning (at the cost of exploration), as agents practice Decentralized Training, Decentralized Execution (DTDE). This is achieved as a collection of learning agents individually experience a commonly parameterized environment, and interact only implicitly via observations of environment modifications. The environment observations serve as a medium of implicit interaction as the attacker takes a single attack action to modify/poison the blueprint of the victim environment, conditioned on the behaviors of all agents present in the population. Agents, therefore, have some influence on each other, since a failure or stubbornness of one of them has an effect on the next attack presented to the entire population. Drawing inspiration from [7], exploration is made more efficient during testing by assigning an independent, random number generation seed to each agent in the victim population. This increases diversity as well as commitment of the victim agents. The Implicit Collective setting can find application in the development of an adaptive single-player computer game engine wherein the adaptive game engine represents an attacker that strives to simultaneously push a complete population of victims (players) towards a target behavior (with maximum player stickiness) and each attack action represents a new release/version of the game.

In Collective and Swarm Collective settings the agents occupy the same copy of the victim environment. Herein, each agent observes (anonymized version of) every other agent, and takes actions conditioned on this observation (along with its own position inside the victim environment). The agents practice Centralized Training, Decentralized Execution (CTDE) where a single network is trained using all victims' interactions. As the agents learn from each other's experiences, these settings support social learning. In Collective setting, individual learning is enhanced by inculcation of "agent indication" i.e. addition of an indication of the observing agent to its observations [24]. This enables the same neural network to represent diverse victim behaviors. In addition, each agent is committed to a separate, diverse behavior by assigning it an independent, random number generation seed [7]. Collective setting, therefore, presents a balanced scenario that houses high support for both individual and social learning. On the other hand, Swarm Collective does not inculcate agent indication and independent random seeds, and therefore strongly promotes social learning. The centrally trained neural network in this setting learns a single optimal victim behavior using all victims' interactions. Lastly, a soft competition is injected into both settings by terminating victim-training episodes as soon as at least one victim reaches the goal state (or maximum training time

has elapsed); to enable faster adoption of optimal behaviors inside the victim population.

3 METHODOLOGY

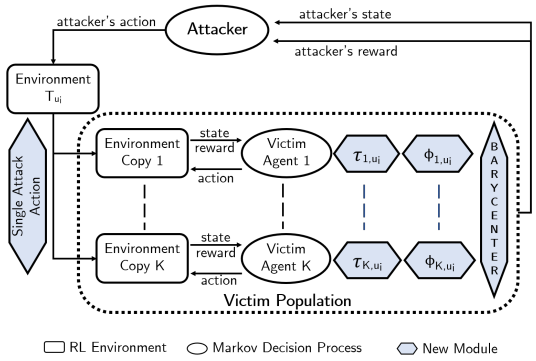

Figure 1: Bi-Level Attack Framework in Implicit Collective Setting

This section describes the developed methodology in increasing levels of detail. First, the overall interaction structure between the attacker and a population of victim learners is presented. Then, the specifics of encoding a distribution of behaviors within a population are described.

3.1 Bi-Level System Architecture

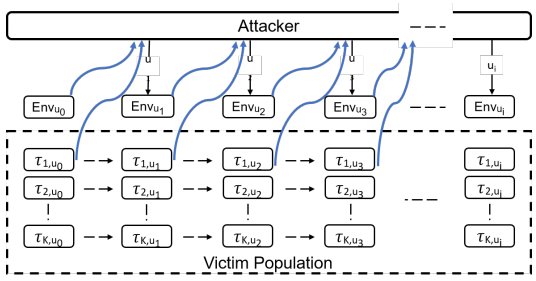

Figure 2: Attack Deployment

We formalize our method as a bi-level hierarchical framework wherein the attacker as well as each member of the victim population is an independent reinforcement-learning agent with its individual learning algorithm, memory, and policy. The victim population is a collection of learners where each member trains to learn the given task in a common environment under the Collective and Swarm Collective settings; and in an environment copy based on the common blueprint, in the Implicit Collective setting. In order to learn the given task, each agent trains to maximize its individual cumulative discounted rewards, which correspond to its individual task. The attacker on the other hand observes the interaction of the population of victims with their environment and, based on the set of observed behaviors, takes an action that modifies the victim environment/blueprint. The goal of the attacker is to sequentially and minimally modify the victim environment dynamics to drive the victim population to adopt the attacker-desired target behavior. Therefore, the overall system is formed by two nested closed-loop learning processes, wherein the attacker and members of the victim population are modeled as Markov Decision Processes (MDPs).

Victim Population MDP: The victim population's Markov process can be denoted by the tuple $< S, A, T_{u_i}, R_v, q_0, \gamma_v >$ where $S = s_1, s_2, ...$, and $A = a_1, a_2, ...$ are the victims' states and actions respectively; $R_v : S \times A \times S \rightarrow \mathbb{R}$ is the reward function which encodes each victim's task; $\gamma_v \in (0, 1)$ is the discount factor, $q_0(S)$ is the distribution over initial states; and, $T_{u_i} : S \times A \times S \rightarrow [0, 1]$ is the probabilistic transition function, where u_i denotes the environment parameterization that has resulted from the first i interventions on the environment, by the attacker. In particular, T_{u_0} refers to the original, unaltered dynamics of the victim environment. A single attack action modifies the victim environment dynamics for the entire victim population. The objective of each member of the victim population is to find an optimal policy within the experienced environment.

Attacker MDP: In our ultra-blackbox setting, the attacker's Markov process is partially observable in nature as the attacker does not have access to the victim policies; and instead, constructs an approximation of the victim population's behavior. However, explicitly solving a POMDP is computationally expensive. Taking inspiration from Belief MDPs [9], this work reduces the attacker POMDP to an MDP. The attacker observation is treated as the attacker state and any underlying uncertainty is absorbed by the stochastic transition function. The attacker's Markov process can be represented by the tuple $< \Theta, U, F, R_a, \tau^*, \gamma_a >$, where: $\Theta = [T_{u_i}, \overline{\phi}_{u_i}]$ is the attacker's state space comprising the victim environment dynamics, T_{u_i} and the victim population's behavior, $\overline{\phi}_{u_i}$ that emerged in response to those dynamics; U is the attacker's action space, i.e., the set of all permissible changes that can be applied to the victim environment dynamics, such that action u_i when applied on the environment with dynamics $T_{u_{i-1}}$ results in an environment with dynamics T_{u_i}. It is important to note here that environment dynamics at attack time step i are a result of accumulated changes caused by attack actions $u_1, u_2, ..., u_i$. $F : \Theta \times U \times \Theta \rightarrow [0, 1]$ is the probabilistic transition function that describes the response of the victim population to environmental experiences, i.e., how the distribution of behaviors within the population changes in response to changes in the environment dynamics; $R_a : \Theta \times U \times \Theta \rightarrow \mathbb{R}$ is the attacker reward function that describes the combined efficiency (how concentrated is the victim population behavior distribution around the ideal behavior, τ^*) and effort of (how small is) the accumulated environment modifications; where τ^* is the attacker-desired target victim policy. The attacker optimizes these dual objectives of efficiency (maximize) and effort (minimum) to learn the best attack generation strategy of the form $\sigma : \Theta \rightarrow U$, $\sigma(u_i|\Theta_{i-1})$. I.e., the attacker seeks the most *efficient* way to force all individual behaviors within the victim population to converge to the target policy τ^*.

3.2 Population Behavior Representation

Development of an attack generation function that is capable of pushing victim populations of different sizes towards a target behavior using a single, constrained attack action at every attack time step, under Blackbox and Ultra-Blackbox settings, entails two major challenges, as mentioned in Section 1.

The first challenge is an accurate approximation of individual behaviors present inside the victim population. Due to the Blackbox/Ultra-Blackbox nature of settings, individual victim behaviors can only be approximated (through observation of across-policy behavior traces) and never be captured completely with 100% certainty. The second challenge is to make the attack generation function agnostic to the size of the victim population.

The attacker observes the actions taken by the victims in different states, as they train to learn their tasks in the victim environment. As the victim population is under training, victims update their internal policies periodically. Depending on the frequency of these updates, each state-action pair of a behavior trace can potentially be generated by a different policy. In this paper, the authors work with a highly interactive setting wherein the victims update their policies after each interaction with their environment. In prior works, the attacker strives to capture a victim's behavior by noting its trajectories in the victim environment and conditioning each attack action on the last trajectory observed prior to the attack action [26, 27]. The last trajectory however on one hand, does not capture information about frequently visited states that were not visited in the last trajectory; and on the other hand, does not retain any information about states/regions that are entirely unvisited and hence unimportant to the victim. Storing multiple recent trajectories can help add information regarding other frequently visited states as well as help capture the stochastic behaviors of a victim. However, in the Blackbox/Ultra-Blackbox settings, it is impossible for the attacker to discern between discarded (old) behaviors and stochastic (current) behaviors of a victim. Moreover, this uncertainty is further exacerbated in multi-agent victim settings wherein victim identities are not stored by the attacker. To do away with this uncertainty, reduce memory requirements, and retain information regarding unvisited (and hence unimportant) states; in this work, the attacker stores the last observed victim action corresponding to each state while a symbolic "no-action" is used to demarcate unvisited states. Information regarding environment configurations that are unimportant with respect to a victim agent's objectives, can prove crucial to the attacker while deciding stealthy and efficient environment modifications that push the complete victim population towards the attacker-desired target behavior. This individual behavior information corresponding to a given victim k will hereafter be denoted as $\tau_{k,u_i} = \{s_1, a_1; s_2, a_2; ...; s_N, a_N\} \forall s_n \in S$, a_n is the latest action taken by victim k in state n or a no-action symbol in case state s_n was never visited by the victim, and N is the total number of states in the given environment with dynamics T_{u_i}. As τ_{k,u_i} contains the latest action / no-action symbol corresponding to all environment configurations, τ_{k,u_i}'s size can become extremely large in high-dimensional environments. Furthermore, in size-agnostic multi-agent victim attacks, the attacker needs a mechanism to generate a uniform-sized representation of all behaviors of the victim population. In this work, these two problems are solved by the attacker by learning a distributional low-dimensional latent space, Φ of individual behaviors using a variational auto-encoder model. The latent behavior distribution corresponding to a given victim's individual behavior τ_{k,u_i} is denoted by ϕ_{k,u_i}. Herein the dimensionality of $\tau_{k,u_i} >> \phi_{k,u_i}$. The variational model consists of an encoder q_e that takes a given victim agent's τ_{k,u_i} as input and outputs parameters to it's latent behavior distribution ϕ_{k,u_i}; and a decoder q_d that takes two inputs, a sample z from the latent distribution ϕ_{k,u_i} and a victim environment state s_n, and outputs the probability with which victim k will take each available action in the given state s_n. The prior distribution $p(z)$ on the latent variables is the standard normal $N(z; 0, I)$ while the evidence lower bound, to be maximized over all k is:

$$\mathbb{E}_{z\sim\phi_{k,u_i}}[\ log\ q_d(a_n|z, s_n)] - D^{kl}\left(\phi_{k,u_i}\ ||\ p(z)\right) \tag{1}$$

The generative capability of the variational individual-behavior model is crucial in solving the second challenge of developing a size-agnostic attack strategy that is transferable across different populations of varied sizes. The authors exploit the regularity of the distributional latent space and utilize the Wasserstein distance [25] to generate a latent distribution that is representative of all individual behaviors approximated from the victim population. Wasserstein distance respects the underlying geometry of the metric space in which the distributions reside. Therefore, unlike other distances like Euclidean, Total Variation, Hellinger, etc, Wasserstein distance provides an aggregation mechanism (Barycenter) that preserves the structure of individual behavior distributions. This property is highly useful for the development of the population behavior representation as it enables the attacker to understand prevalent victim behaviors and thereby learn an efficient attack strategy for the population. Secondly, Wasserstein distance is insensitive to small changes in distributions which is a crucial property for this work as τ_{k,u_i}s are approximate representations of the actual policies of the victims and hence inherently possess a sizable margin of error. Lastly, Wasserstein distances can be computed between two discrete, two continuous as well as a discrete and a continuous distribution. This property ensures the scalability of the developed methodology to large, continuous environments on one hand, and on the other hand, enables the developed approach to be agnostic to the nature (discrete/continuous) of ϕ_{k,u_i}. In this work the authors use a fixed-point approach for fast computation of the Wasserstein barycenter (Fréchet mean), $\overline{\phi}_{u_i}$ [1] of the individual latent behavior distributions ϕ_{k,u_i} corresponding to all K agents present in the victim population; $[\phi_{1,u_i}, \phi_{2,u_i}, ..., \phi_{K,u_i}]$. In formula 2 given below, W stands for L2-Wasserstein distance, and λ corresponds to the importance of a particular latent behavior distribution ϕ_{k,u_i}. Herein λ_k is assigned the value of 1 for all k as the behavior of each victim is equally important for the attacker to consider.

$$\sum_{k=1}^{K} \lambda_k W^2(\phi_{k,u_i}, \overline{\phi}_{u_i}) = \min_{\phi\in\Phi}\left\{\sum_{k=1}^{K} \lambda_k W^2(\phi_{k,u_i}, \phi)\right\} \tag{2}$$

4 EXPERIMENTS

One of the primary cognitive capabilities is the ability to navigate in a new environment. This work tests and establishes the quality of the proposed methodology by training an attacker to learn to attack a population of navigational agents in a stochastic grid environment titled 3D Grid World [21]. This environment simulates an uneven terrain on a grid of 2 dimensional cells. The unevenness corresponds to the 3^{rd} dimension of the grid and is due to the elevation/altitude associated with each grid cell. The relative elevation between two cells decides the transition probabilities between them. A change in this relative elevation thus changes how the environment responds

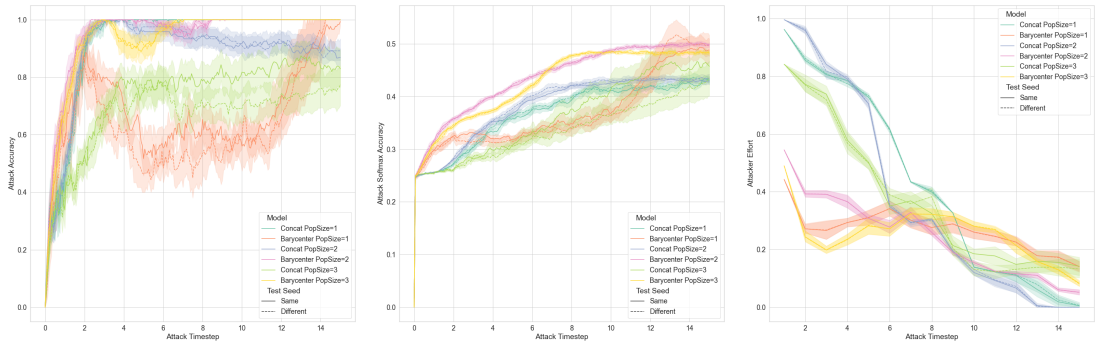

Figure 3: Accuracy, Softmax Accuracy and Effort of attacks, trained and tested on same sized Implicit Collective victim populations

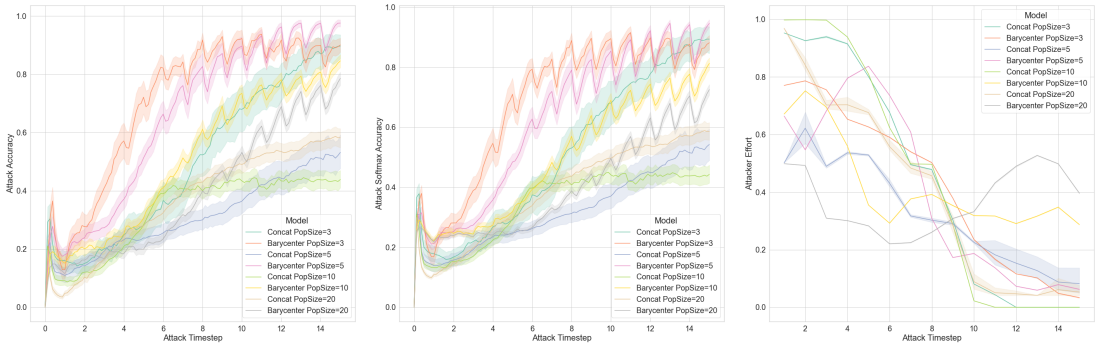

Figure 4: Accuracy, Softmax Accuracy and Effort of attack strategies trained and tested on same sized Collective victim populations

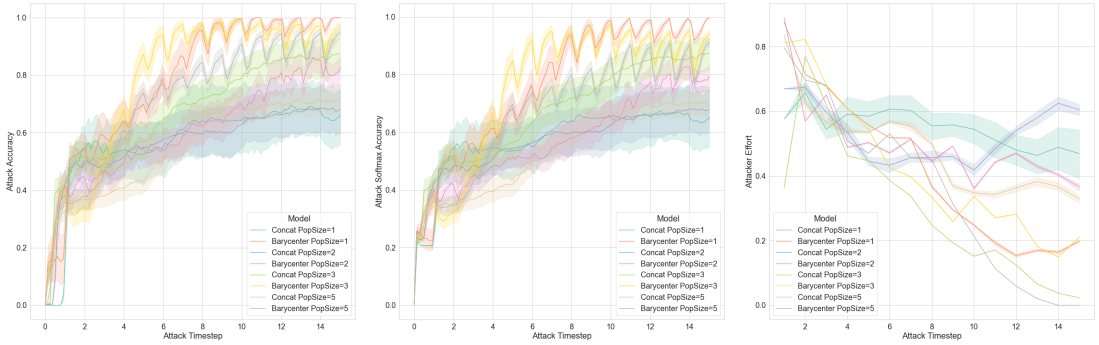

Figure 5: Accuracy, Softmax Accuracy and Effort of attacks, trained and tested on same sized Swarm Collective victim populations

to a navigating agent's actions. The navigating agent's task is to find the optimal path from the start cell to the goal cell and its state is its position inside the grid world in Implicit Collective setting and its position along with the position of all other members of the population, in Collective and Swarm Collective settings. At each time step, the navigating agent observes its state and takes one step in any of the four cardinal directions (N,S,E,W). Each agent receives a reward of -1 for every action it takes until it reaches the goal state. A given victim training episode terminates once all agents in Implicit Collective setting and at least one agent in Collective and Swarm Collective settings, reaches the goal state or maximum time has elapsed. This pushes the agents to find the shortest path(s) to the goal cell. 3D Grid World also allows the presence of an additional agent, the elevation expert who can view the altitude of each grid cell and take a constrained action to modify it. The elevation expert's state space comprises of the grid cells' altitudes along with the navigational population's behavior, while its action space is a vector of real numbers $[x^1, x^2, x^3, ..., x^M], x \in [-1.0, 1.0]$ where M is the

total number of cells in the grid. In this work, each member of the victim population is a navigating agent while the attacker is the elevation expert. The attacker's objective is to efficiently force the victim population to follow a target path to the attacker's desired destination. The target path is not an optimal path in the original environment and thus is not the optimal choice for the victims under default environment dynamics.

The performance of the attacker is measured in terms of the accuracy (Attack Accuracy) and strength (Attack SoftMax Accuracy) with which the victim population (unknowingly) adopts the target behavior as well as the cumulative changes brought about in the victim environment by the attacker (Attacker Effort). **Attack Accuracy (**abbreviated as **@Acc)** computes the level of adoption of the target behavior by the victim. A target state (state included in the target path) is assigned an adoption accuracy of 1.0 if the victim assigns the highest probability to the target action (attacker-desired victim-action) in that state. The final accuracy is the sum of the adoption accuracies of all target states, divided by the number of target states. The convergence rate of this measure reflects how quickly the attack led to target behavior adoption in the population. **Attack SoftMax Accuracy (@SoftAcc)** computes the probability assigned to the target path by the victim and is computed as the sum of the target actions' probabilities in the target states, divided by the number of target states. This measure reflects the strength with which the population adopts the target behavior. **Attacker Effort (@Effort)** computes the degree to which the attacker has modified the environment and is computed as the mean of the absolute difference between the previous and current attack time step's grid cells' altitudes. The plots of these three measurements are provided for each experiment conducted in this study. The attacker training episodes are 15-step sequential attacks on freshly initialized victim populations wherein attack step 0 corresponds to the original environment with default dynamics. After each episode, the attack strategy employed in that episode is saved if it is better or equal to the best attack strategy found so far. These best strategies are selected by prioritizing @Acc, because the main goal of this work is to find strategies that push victim populations to adopt the target behavior, while, the amount of changes made to the environment in order to achieve this (@Effort) is the lower priority objective, and the strength of target behavior adoption (@SoftAcc) demonstrates additional capabilities of the attack strategies. @Acc and @SoftAcc are measured along the victim timescale to observe how the accuracies change (and thereby understand how the victim population behaves) in-between attack actions. @Effort on the other hand can only be measured corresponding to each attack action and hence is measured along the attacker timescale. Due to this difference, accuracy plots begin from attack step 0 while effort plots begin from attack step 1. Each graph corresponds to attacks carried out on 20 separate victim populations.

Given that this is the first work that studies a multi-victim attack via a common adversarial action, we have constructed a high-performance artificial baseline for comparison by extending the single-agent SOTA environment-poisoning model, TEPA [27] to the multi-agent setting. TEPA is an auto-encoder-based model that is shown capable of extracting the behavior of a single victim agent in whitebox and proxy-blackbox adversarial settings. TEPA uses state, action trajectories to represent the single victim's behavior. In this work, TEPA becomes capable of attacking a population by utilizing the concatenation of latent representations of individual behavior trajectories as the representation of the population's behavior. This population behavior representation includes all available information from individual behavior representations, in its entirety, which on one hand empowers the attacker with more information but on the other hand, does not support learning of size-agnostic and/or transferable attacks. This method of concatenation is utilized for comparison in order to demonstrate the performance of an attack that makes use of the maximum available information compared to one that utilizes less information but has size-agnostic capabilities. Two experiments are used to aid this comparison under each multi-victim setting. In experiment A - *Behavior Concatenation vs Barycenter*, attack strategies are trained and tested on populations of same size. Each strategy is tested on 20 populations. In Implicit Collective setting with Q-learning victim agents, 10 test populations use the same seed as the one used by the victim populations during training, while each agent in each of the remaining 10 populations uses a different seed. In Collective and Swarm Collective settings with DQN victim agents, neural networks corresponding to 10 test populations are initialized using random numbers from the same range as the range used for initializing populations during training, while the remaining 10 test populations are initialized using a different range. Experiment B - *Size Agnosticity of Behavior Barycenter* demonstrates the size-agnostic and transferable capabilities of the developed barycenter-based approach. The attacker learns attack strategies on populations of sizes 3,5,10, and 20. Each of these strategies is tested on 4 sets of 20 populations of sizes 3,5,10, and 20 respectively. Each set of 20 populations is created in the same manner as described above.

Experiment A under Implicit Collective setting presented in Figure 3 encapsulates the feasibility study of this work wherein the proposed and artificially-constructed baseline methods are trained to attack populations of sizes 1,2, and 3. The concatenation-based baseline converges to 1.0 @Acc within 3 attack actions when trained to attack a population of size 1. However, as the size of the victim population increases, mean @Acc decreases with an increase in variance. Moreover, @Acc of attacks on size-3 populations, initialized using different seeds is much lower than attacks initialized using the same seed. This shows that the baseline attack strategy finds it harder to attack larger populations as well as populations with greater differences from the ones used during training, suggesting that complete information regarding multiple victim trajectories confuses the attacker, especially when those trajectories correspond to differently initialized victim agents. Population behavior barycenter attacks on the other hand display the opposite trend. The attack strategy trained on populations of size 1 performs worse than those trained on populations of sizes 2 and 3, in terms of rate of convergence to 1.0 @Acc as well as its variance. Also, interestingly, attacks on populations initialized with different seeds converge slightly faster than those initialized with the same seed. @SoftAcc graph shows that barycenter-based attacks cause stronger adoption of target behavior as the victim population assigns higher probabilities to attacker-desired actions. On the other hand, @Effort graph shows that at the beginning of the attack, concatenation-based strategies modify nearly all cells by almost maximum permissible amount exerting @Effort between 0.8 and 1.0; while Barycenter-based approaches

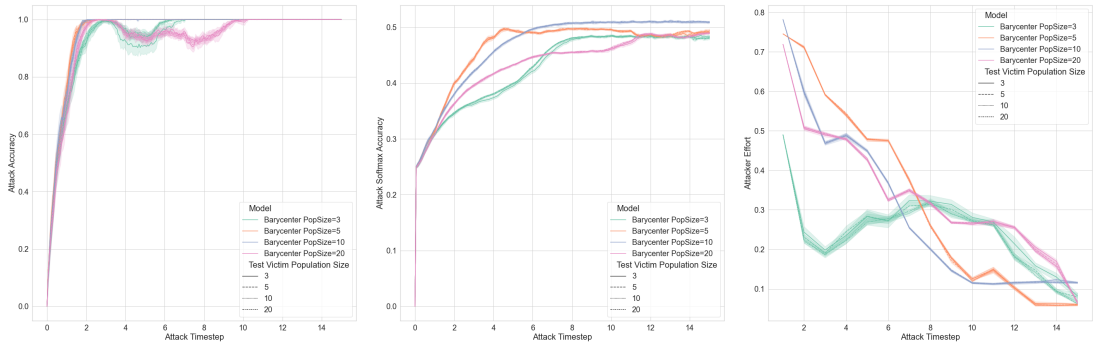

Figure 6: Accuracy, Softmax Accuracy and Effort of Barycenter attacks, tested on Implicit Collective victims of sizes 3, 5, 10 and 20

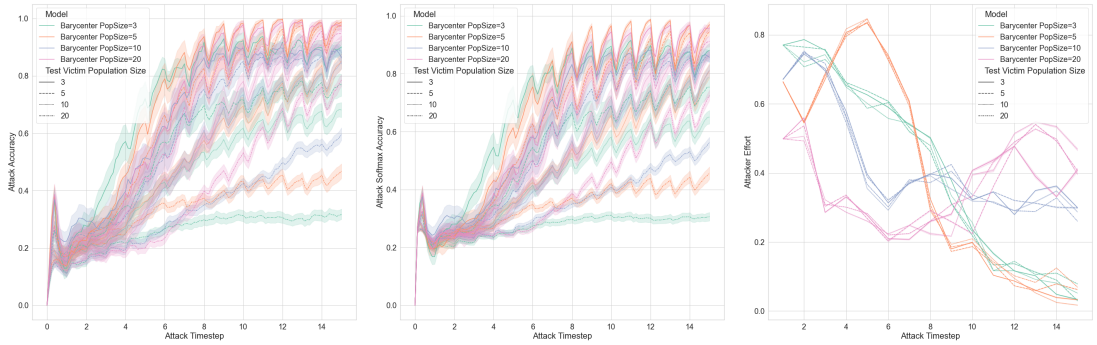

Figure 7: Accuracy, Softmax Accuracy and Effort of Barycenter attacks, tested on Collective victim populations of sizes 3, 5, 10 and 20

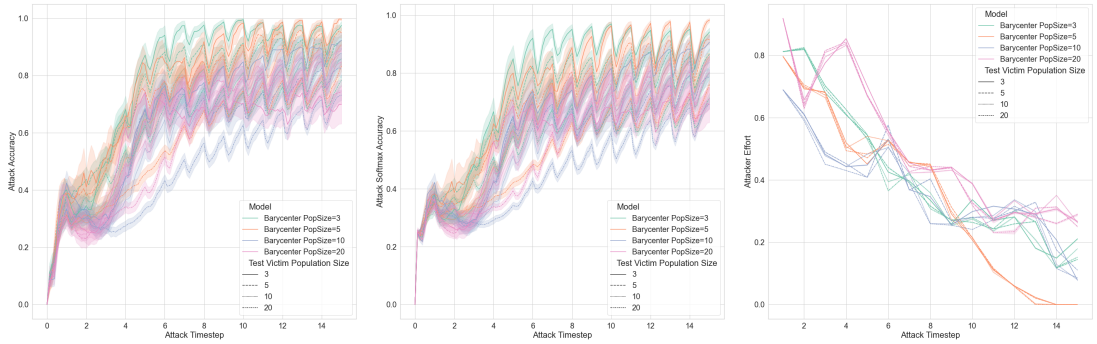

Figure 8: Accuracy, Softmax Accuracy and Effort of Barycenter attacks, tested on Swarm Collective victims of sizes 3, 5, 10 and 20

begin the attack at approximately half the level of modifications (between 0.4 and 0.6). By the end of the attack, all strategies are able to bound environment modifications between 0.0 and 0.2. This experiment, therefore, shows the feasibility of multi-victim attacks by demonstrating successful and efficient attacks on different-sized Implicit Collectives.

Experiment A under Collective setting presented in Figure 4 serves to test the capabilities of the proposed and artificially-constructed baseline methods, by training attacks against populations of sizes 3, 5, 10, and 20. In this setting, the baseline achieves high @Acc only when trained to attack small 3-agent populations. On the other hand, barycenter-based attacks on all population sizes achieve high @Acc by the end of the attack. @Acc achieved by the last attack time step decreases with increasing population size but remains above 0.8 throughout. It is interesting to note that @SoftAcc also displays the same trends as @Acc. These results imply that barycenter-based strategies not only push the victim population to assign the maximum probability to attacker-desired target actions but also ensure that this probability is high (roughly between 0.7 and 1.0) by the end of the attack. @Effort of most concatenation-based strategies begins

at high values (near 1.0) and reaches low values (below 0.1) by the end of the attack. Whereas, @Effort of barycenter-based strategies begins at medium values (between 0.5 and 0.8) and reaches low values (between 0.0 and 0.1) for smaller populations and medium-low values (between 0.3 and 0.4) for larger populations by the end of the attack. Therefore, with increasing population size, concatenation-based approach minimizes effort without achieving high accuracy. In contrast, barycenter-based approach finds strategies that continue to exert some effort by the end of the attack in order to achieve high accuracy. In addition, it is interesting to note that all barycenter-based attacks in Collective setting demonstrate a climbing zig-zag behavior in terms of @Acc and @SoftAcc. During this climbing zig-zag behavior, the attack accuracies suffer a dip as soon as an attack action poisons the environment but eventually reaches the largest value seen so far; before the next poisoning takes place. Barycenter-based approach, therefore, seems to possess the capability of finding involved/complex attack strategies that choose to suffer short-term losses in order to obtain long-term gains.

Figures 6 and 7 present experiment B under Implicit Collective and Collective settings respectively. In Implicit Collective setting test performance across all three measures are agnostic to the testing population size and all barycenter-based strategies converge to the perfect @Acc of 1.0, high @SoftAcc of 0.5, and low @Effort below 0.15, by the end of the attack. In Collective setting, for each strategy, accuracy decreases with increasing test population size. This decrease however reduces with increasing training population size. Barycenter-based approach, therefore, is not completely size-agnostic under Collective setting, especially w.r.t strategies trained on very small populations and tested on large populations, but performance degrades gracefully.

The Swarm Collective setting is utilized to further test the capabilities of the proposed barycenter-based method. In this setting, as shown in Figure 5, the majority of both concatenation and barycenter based strategies achieve better accuracies while exerting higher effort, compared to their Collective setting counterparts. All baseline strategies except the strategy trained on size-3 populations achieve high @Acc and SoftAcc between 0.6 and 0.8, in contrast to the Collective setting, wherein they achieve @Acc and @SoftAcc between 0.4 and 0.6, by the end of the attack. Size-3 baseline strategy achieves high @Acc *and* @SoftAcc (0.9) by the end of the attack in both Collective and Swarm Collective settings. Higher accuracies for Swarm Collectives are accompanied by higher @Effort as all strategies except size-3 strategy exert effort between 0.3 and 0.6, in contrast to Collective setting wherein they reduce @Effort to below 0.2 by the end of the attack. Size-3 strategy in both Collective and Swarm Collective settings converge towards 0.0 @Effort by the end of the attack. This shows that the baseline method that uses concatenation of trajectory-based individual victims' behaviors to capture the population behavior is less capable of accurately understanding population behavior of populations of sizes $\neq 3$. Barycenter-based strategies that achieve high @Acc of above 0.8 and high @SoftAcc of above 0.7 in Collective setting achieve even higher accuracies under Swarm Collective setting with majority of the strategies reaching above 0.9 @Acc and @SoftAcc by the end of the attack. This performance boost comes with slightly higher @Effort which is between 0.2 and 0.4 for Swarm Collectives compared to between 0.0 and 0.4 in the Collective setting. Lastly, like Collective setting, barycenter-based attacks also exhibit the climbing zig-zag behavior w.r.t. @Acc and @SoftAcc under Swarm Collective setting. Experiment B under Swarm Collective setting presented in Figure 8 shows that strategies trained on smaller populations and tested on larger populations perfectly retain the climbing zig-zag behavior but this behavior slightly degrades for strategies trained on larger and tested on smaller populations. @SoftAcc shows similar trends to @Acc while @Effort remains largely unchanged across tests on different sized test populations. Therefore, barycenter-based approach under Swarm Collective setting is highly size-agnostic in nature.

These results imply that barycenter-based strategies are more effective at attacking collectives that practice individual (Implicit Collective) or social (Swarm Collective) learning than at attacking collectives that exploit both (Collective).

5 CONCLUSION AND FUTURE WORK

This paper develops an extension of environmental poisoning attacks to populations of reinforcement (RL) agents and introduces the Collective Environment Poisoning (CEP) framework that constitutes; a) Implicit Collective (Blackbox); b) Collective (Ultra-Blackbox); and c) Swarm Collective (Ultra-Blackbox) settings. The authors show that concatenation-based population behavior representation, not only creates attack strategies that are non-transferable to different-sized populations but also overloads the attacker with information inhibiting it from finding strategies that ensure high attack accuracy with low effort. In contrast, barycenter-based population behavior representation achieves both of the aforementioned feats in Implicit Collective setting wherein population members practice individual learning. In Collective and Swarm Collective settings wherein victim agents learn via individual+social and social learning respectively, barycenter-based attack strategies achieve high accuracies with bounded effort when trained and tested on same-sized populations. These strategies are transferable to different-sized populations except for strategies trained on very small populations ($\sim$3) and tested on very large populations ($\sim$20). However, even in such cases, the performance degrades gracefully.

The current framework approximates each individual victim's policy using the last action taken by the victim in each environment state. This data can however only be captured for victims training in a discrete environment. Similarly, KLR that is used to create the extrinsic (Blackbox) and intrinsic (Ultra-Blackbox) attacker reward signals requires the underlying MDP to be based on an environment with discrete state and action spaces. Our next step entails expansion of the proposed methodology to continuous environments, e.g., by utilizing discrete latent space encoders. Moreover, the CEP framework does not include the provision of attacking open multi-victim systems wherein victims can freely enter/exit the system or heterogeneous victim populations. Future work constitutes expanding CEP to study attacks on heterogeneous, open multi-victim systems. Ultimately our goal is to understand these attacks so as to develop defense mechanisms and contribute towards safe and robust RL solutions.

ACKNOWLEDGMENTS

This research was supported in part by the NTU SUG "Choice manipulation and Security Games".

REFERENCES

[1] P. C. Álvarez-Esteban, E. Del Barrio, J. A. Cuesta-Albertos, and C. Matrán. 2016. A fixed-point approach to barycenters in Wasserstein space. *J. Math. Anal. Appl.* 441, 2 (2016), 744–762.

[2] V. Behzadan and A. Munir. 2017. Vulnerability of deep reinforcement learning to policy induction attacks. In *MLDM*. 262–275.

[3] P. Chelarescu. 2021. Deception in Social Learning: A Multi-Agent Reinforcement Learning Perspective. *arXiv preprint arXiv:2106.05402* (2021).

[4] T. Chen, J. Liu, Y. Xiang, W. Niu, E. Tong, and Z. Han. 2019. Adversarial attack and defense in reinforcement learning-from AI security view. *Cybersecurity* 2, 1 (2019), 1–22.

[5] Y. Chen, Z. Zheng, and X. Gong. 2022. MARNet: Backdoor Attacks Against Cooperative Multi-Agent Reinforcement Learning. *IEEE Transactions on Dependable and Secure Computing* (2022).

[6] Nathaniel D Daw, Yael Niv, and Peter Dayan. 2005. Uncertainty-based competition between prefrontal and dorsolateral striatal systems for behavioral control. *Nature neuroscience* 8, 12 (2005), 1704–1711.

[7] M. Dimakopoulou, I. Osband, and B. Van Roy. 2018. Scalable coordinated exploration in concurrent reinforcement learning. In *NeurIPS*.

[8] I. Ilahi, M. Usama, J. Qadir, M. U. Janjua, A. Al-Fuqaha, D. T. Hoang, and D. Niyato. 2021. Challenges and countermeasures for adversarial attacks on deep reinforcement learning. *IEEE Transactions on Artificial Intelligence* 3, 2 (2021), 90–109.

[9] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. 1998. Planning and acting in partially observable stochastic domains. *Artificial Intelligence* 101, 1-2 (1998), 99–134.

[10] G. Lample and D. S. Chaplot. 2017. Playing FPS games with deep reinforcement learning. In *AAAI*.

[11] J. Lin, K. Dzeparoska, S. Q. Zhang, A. Leon-Garcia, and N. Papernot. 2020. On the robustness of cooperative multi-agent reinforcement learning. In *2020 IEEE Security and Privacy Workshops (SPW)*. 62–68.

[12] T. Liu, J. McCalmon, M. A. Rahman, C. Lischke, T. Halabi, and S. Alqahtani. 2023. Weaponizing Actions in Multi-Agent Reinforcement Learning: Theoretical and Empirical Study on Security and Robustness. In *PRIMA*. 347–363.

[13] A. Lupu and D. Precup. 2020. Gifting in multi-agent reinforcement learning. In *AAMAS*. 789–797.

[14] B. Marthi, S. Russell, D. Latham, and C. Guestrin. 2005. Concurrent hierarchical reinforcement learning.. In *IJCAI*. 779–785.

[15] G. Mialon, D. Chen, A. d'Aspremont, and J. Mairal. 2020. A trainable optimal transport embedding for feature aggregation. In *ICLR*.

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

[17] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. 2015. Human-level control through deep reinforcement learning. *nature* 518, 7540 (2015), 529–533.

[18] E. Parisotto, S. Ghosh, S. B. Yalamanchi, V. Chinnaobireddy, Y. Wu, and R. Salakhutdinov. 2019. Concurrent meta reinforcement learning. *arXiv preprint arXiv:1903.02710* (2019).

[19] N. Pham, L. Nguyen, J. Chen, L. T. Hoang, S. Das, and L. Weng. 2022. c-MBA: Adversarial Attack for Cooperative MARL Using Learned Dynamics Model. In *ML Safety WS @ NeurIPS*.

[20] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*. 652–660.

[21] Z. Rabinovich, L. Dufton, K. Larson, and N. R. Jennings. 2010. Cultivating Desired Behaviour: Policy Teaching Via Environment-Dynamics Tweaks. In *AAMAS*. 1097–1104.

[22] A. Rakhsha, G. Radanovic, R. Devidze, X. Zhu, and A. Singla. 2020. Policy teaching via environment poisoning: Training-time adversarial attacks against reinforcement learning. In *ICML*. PMLR, 7974–7984.

[23] K. Skianis, G. Nikolentzos, S. Limnios, and M. Vazirgiannis. 2020. Rep the set: Neural networks for learning set representations. In *AIStats*. 1410–1420.

[24] Justin K Terry, Nathaniel Grammel, Ananth Hari, Luis Santos, and Benjamin Black. 2020. Revisiting parameter sharing in multi-agent deep reinforcement learning. *arXiv preprint arXiv:2005.13625* (2020).

[25] L. N. Vaserstein. 1969. Markov processes over denumerable products of spaces, describing large systems of automata. *Problemy Peredachi Informatsii* 5, 3 (1969), 64–72.

[26] H. Xu, X. Qu, and Z. Rabinovich. 2022. Spiking Pitch Black: Poisoning an Unknown Environment to Attack Unknown Reinforcement Learners. In *AAMAS*. 1409–1417.

[27] H. Xu, R. Wang, L. Raizman, and Z. Rabinovich. 2021. Transferable Environment Poisoning: Training-time Attack on Reinforcement Learning. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*. 1398–1406.

[28] Anil Yaman, Nicolas Bredeche, Onur Çaylak, Joel Z Leibo, and Sang Wan Lee. 2022. Meta-control of social learning strategies. *PLoS computational biology* 18, 2 (2022), e1009882.

[29] J. Yang, A. Li, M. Farajtabar, P. Sunehag, E. Hughes, and H. Zha. 2020. Learning to incentivize other learning agents. In *NeurIPS*. 15208–15219.

[30] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola. 2017. Deep sets. In *NeurIPS*.