Competitive Multi-agent Inverse Reinforcement Learning with Sub-optimal Demonstrations

Xingyu Wang, Diego Klabjan
Department of Industrial Engineering and Management Sciences
Northwestern University, Evanston, IL, 60208
`xingyuwang2017@u.northwestern.edu, d-klabjan@northwestern.edu`

May 24, 2018

A Appendix

A.1 Notations, Characterization of Nash Equilibriums in zero-sum discounted stochastic games

We first give a formal definition of Q functions and advantage functions since they are involved in the discussion below. Aside from the state value function $v^{f,g}(s;R)$, another useful metric is the state-action value function (namely, Q-function)

$$Q_f^{f,g}(s,a;R) = R(s) + \gamma \mathbb{E}_{a^g \sim g(a|s),\ s' \sim P(s'|s,a,a^g)} \Big[v^{f,g}(s';R) \Big],$$
$$Q_g^{f,g}(s,a;R) = -R(s) + \gamma \mathbb{E}_{a^f \sim f(a|s),\ s' \sim P(s'|s,a^f,a)} \Big[- v^{f,g}(s';R) \Big].$$

Based on the state value function and Q-function, the advantage function measuring benefits of actions over current policies is defined as

$$A_f^{f,g}(s,a^f;R) = Q_f^{f,g}(s,a^f;R) - v^{f,g}(s;R),$$
$$A_g^{f,g}(s,a^g;R) = Q_g^{f,g}(s,a^g;R) + v^{f,g}(s;R).$$

The advantage functions can be used for characterization of optimal policies in both single-agent and multi-agent MDPs. In the zero-sum case, we know that a Nash Equilibrium $(f^*(R), g^*(R))$ ensures that, for any $s \in S$ and any policy pair (f,g) we have

$$v^{f^*(R),g}(s;R) \geq v^{f^*(R),g^*(R)}(s;R) \geq v^{f,g^*(R)}(s;R).$$

Therefore, $f^*(R)$ is a solution to optimization problem

$$\max_f \min_g \frac{1}{N_s} \sum_{s\in S} v^{f,g}(s;R),$$

and $g^*(R)$ is a solution to

$$\min_g \max_f \frac{1}{N_s} \sum_{s\in S} v^{f,g}(s;R).$$

As already mentioned above, we can characterize Nash Equilibriums in zero-sum discounted games using advantage functions. To be specific, $\big(f^*(R), g^*(R)\big)$ is a Nash Equilibrium if and only if for any s,

$$A_f^{f^*(R),g^*(R)}(s,a;R) \leq 0 \text{ for } \forall a \in \mathcal{A}^f, \text{ and } A_f^{f^*(R),g^*(R)}(s,a;R) = 0 \text{ if } f^*(R)(a|s) > 0;$$
$$A_g^{f^*(R),g^*(R)}(s,a;R) \leq 0 \text{ for } \forall a \in \mathcal{A}^g, \text{ and } A_g^{f^*(R),g^*(R)}(s,a;R) = 0 \text{ if } g^*(R)(a|s) > 0.$$

A.2 Details of the Nash Equilibrium Algorithm

Two deep neural networks $f_{\theta_f}(s)$ and $g_{\theta_g}(s)$ are used, parametrized by θ_f and θ_g respectively. Both networks take state vector $s \in \mathcal{S}$ as input, which is completely public to both sides. Each network outputs a probability distribution over action space, namely

$$f_{\theta_f}(s) = \Big(\ f_{\theta_f}(a^f(1) \mid s), \ f_{\theta_f}(a^f(2) \mid s), \ \ldots, \ f_{\theta_f}(a^f(|\mathcal{A}^f|) \mid s) \ \Big),$$

$$g_{\theta_g}(s) = \Big(\ g_{\theta_g}(a^g(1) \mid s), \ g_{\theta_g}(a^g(2) \mid s), \ldots, \ \ g_{\theta_g}(a^g(|\mathcal{A}^g|) \mid s) \ \Big).$$

Agents then sample actions from the probability distribution and act accordingly.

As for policy gradient methods used in our algorithm, we choose the actor-critic style Proximal Policy Optimization algorithm (PPO) from [1] because of its superior and stable performances. To perform actor-critic style PPO, a state value model is required and we denote it as $v^{f,g}_{\theta_v}(s)$. Again, $v^{f,g}_{\theta_v}(s)$ is a deep neural network that takes state vector s as input, and outputs a scalar as the estimation for state value defined in (1).

The online training on f and g relies on T-step trajectories of the game. From a randomly initialized s_0, we run the agents for T consecutive steps to obtain a trajectory as an ordered tuple

$$\Big(s_0, s_1, \ldots, s_{T-1}, s_T\Big).$$

At each state s_t, a^f_t, a^g_t are the actions taken by f, g based on the incumbent policy parameters, R_t is the immediate reward received by f at step t, and s_{t+1} is the next state visited by agents after their actions.

Similar to [1], a set of target networks is maintained for generating trajectories, while the training step based on observed trajectories updates the original networks. Parameters for target networks are denoted as $(\theta^{\text{target}}_f, \theta^{\text{target}}_g, \theta^{\text{target}}_{v,f}, \theta^{\text{target}}_{v,g})$. After every K_{refresh} iterations, they are periodically refreshed by $(\theta_f, \theta_g, \theta_{v,f}, \theta_{v,g})$ we are training. Estimation of advantage for f at each step is

$$\hat{A}^f_t = \delta^f_t + (\gamma\lambda)\delta^f_{t+1} + ... + (\gamma\lambda)^{T-t-1}\delta^f_{T-1} \ \text{ for } \ t = 0, 1, 2, ..., T-1,$$
$$\delta^f_t = R_t + \gamma v^{f,g}_{\theta^{\text{target}}_{v,f}}(s_{t+1}) - v^{f,g}_{\theta^{\text{target}}_{v,f}(s_t)},$$

and the clipped loss in PPO for f is

$$L^{f,\text{CLIP}}(\theta_f) = -\sum_t \min\Big(r^f_t(\theta_f)\hat{A}^f_t, (1-\epsilon)\hat{A}^f_t, (1+\epsilon)\hat{A}^f_t\Big),$$
$$r^f_t(\theta_f) = \frac{f_{\theta_f}(a^f_t|s_t)}{f_{\theta^{\text{target}}_f}(a^f_t|s_t)}.$$

Similarly, we formulate the clipped loss for g as

$$\hat{A}^g_t = \delta^g_t + (\gamma\lambda)\delta^g_{t+1} + ... + (\gamma\lambda)^{T-t-1}\delta^g_{T-1} \ \text{ for } \ t = 0, 1, 2, ..., T-1,$$
$$\delta^g_t = -R_t + \gamma v^{f,g}_{\theta^{\text{target}}_{v,g}}(s_{t+1}) - v^{f,g}_{\theta^{\text{target}}_{v,g}}(s_t),$$

$$L^{g,\text{CLIP}}(\theta_g) = -\sum_t \min\Big(r^g_t(\theta_f)\hat{A}^g_t, (1-\epsilon)\hat{A}^g_t, (1+\epsilon)\hat{A}^g_t\Big),$$
$$r^g_t(\theta_g) = \frac{g_{\theta_g}(a^g_t|s_t)}{g_{\theta^{\text{target}}_g}(a^g_t|s_t)}.$$

The loss for updating $\theta_{v,f}, \theta_{v,g}$ is

$$L^v(\theta_{v,f}, \theta_{v,g}) = \sum_t \left[(v^{f,g}_{\theta_{v,f}}(s_t) - v^{f,\text{target}}_t)^2 + (v^{f,g}_{\theta_{v,g}}(s_t) - v^{g,\text{target}}_t)^2 \right],$$
$$v^{f,\text{target}}_t = v^{f,g}_{\theta^{\text{target}}_{v,f}}(s_t) + \hat{A}^f_t,\ v^{g,\text{target}}_t = v^{f,g}_{\theta^{\text{target}}_{v,g}}(s_t) + \hat{A}^g_t.$$

The full algorithm is shown below. As we have mentioned, the proposed algorithm is similar to GAN since g is updated at most of the iterations while in every K_{cycle} iterations only a small proportion of them are meant for f step. Though we do not show this in the algorithm, we recommend the use of a batch of agents running in parallel for collecting gradients (as in [1]). Lastly, we reiterate that the algorithm we present is meant for finding $f^*(R)$. The training for g^* is conducted separately in a similar fashion.

Algorithm 1 Adversarial Training Algorithm for Solving $f^*(R)$ in Zero-Sum Games (Complete)

```
Require:
    Integers $K_g, K_{\text{cycle}}, K_{\text{refresh}}$; learning rates $\lambda_g$, $\lambda_f$; horizon length $T$.

 1: Initialize:  Parameters $\theta_f, \theta_g$ for policy models and $\theta_v$ for state value model.
 2: Set $\theta_f^{\text{target}} \leftarrow \theta_f$, $\theta_g^{\text{target}} \leftarrow \theta_g$, and $\theta_v^{\text{target}} \leftarrow \theta_v$ .
 3: for $i = 1, 2, 3, ....$ do
 4:     Randomly initialize the starting state $s_0$.
 5:     From initial state $s_0$, run $f_{\theta_f^{\text{target}}}(a|s)$, $g_{\theta_g^{\text{target}}}(a|s)$ for $T$ steps
 6:     Calculate estimated advantages for player $f$ and $g$: $\hat{A}_0^{f/g}, \hat{A}_1^{f/g}, ..., \hat{A}_{T-1}^{f/g}$.
 7:     if $i$ % $K_{\text{cycle}} \leq K_g$ then                                    ▷ $g$ step
 8:         $\theta_g \leftarrow \theta_g - \lambda_g \nabla_\theta L^{g,\text{CLIP}}(\theta)|_{\theta=\theta_g}$
 9:         $\theta_{v,f} \leftarrow \theta_{v,f} - \lambda_f \nabla_\theta L^v(\theta, \theta')|_{\theta=\theta_{v,f}, \theta'=\theta_{v,g}}$
10:         $\theta_{v,g} \leftarrow \theta_{v,g} - \lambda_g \nabla_{\theta'} L^v(\theta, \theta')|_{\theta=\theta_{v,f}, \theta'=\theta_{v,g}}$
11:     else                                                              ▷ $f$ step
12:         $\theta_f \leftarrow \theta_f - \lambda_f \nabla_\theta L^{f,\text{CLIP}}(\theta)|_{\theta=\theta_f}$
13:     if $i$ % $K_{\text{refresh}}$ $= 0$ then                         ▷ refresh target networks
14:         Set $\theta_f \leftarrow \theta_f^{\text{target}}$, $\theta_g \leftarrow \theta_g^{\text{target}}$, $\theta_{v,f} \leftarrow \theta_{v,f}^{\text{target}}$ , and $\theta_{v,g} \leftarrow \theta_{v,g}^{\text{target}}$ .
```

A.3 Proof of Proposition 1

Proposition 1: Function $F(f)$ has no local maxima. Namely, if at a certain $\tilde{f}$ there exists no feasible strictly ascent direction for $F(\tilde{f})$, then

$$F(\tilde{f}) = \max_{f \in \mathcal{F}} F(f).$$

Proof. Without loss of generality we can assume that there exists a state s_0 under which both agents receive zero rewards, and regardless of the actions the agents take, any $s \in \mathcal{S}$ has the same probability to be visited at the next step. Formally,

$$\begin{aligned} R(s_0) &= 0, \\ p(s'|s_0, a^f, a^g) &= \frac{1}{N_s} \text{ for } \forall s' \in \mathcal{S}, \forall a^f \in \mathcal{A}_f, \forall a^g \in \mathcal{A}_g. \end{aligned} \tag{1}$$

Given policies f, g and the discount factor $\gamma \in (0,1)$, we have $v^{f,g}(s_0) = \frac{1}{\gamma N_s} \sum_{s \in \mathcal{S}} v^{f,g}(s)$. Therefore,

$$F(f) = \frac{1}{\gamma} \min_g v^{f,g}(s_0).$$

We hereby use this artificial starting state s_0 to simplify the notation. The conclusions below are adapted to the multi-agent setting in our case from the reinforcement learning perspective.

The policy gradient theorem (a variant of (13.5) in [2]) where the agents employ policies f and g, the starting state is s_0, and policy f is a function f_θ parametrized by θ reads

$$\begin{aligned} \nabla_\theta v_f^{f,g}(s_0) &= \sum_s d^{f,g}(s) \sum_{a^f} Q_f^{f,g}(s, a^f) \nabla_\theta f(a^f|s) \\ &= \sum_s d^{f,g}(s) \sum_{a^f} A_f^{f,g}(s, a^f) \nabla_\theta f(a^f|s) + \sum_s d^{f,g}(s) \sum_{a^f} v^{f,g}(s) \nabla_\theta f(a^f|s) \\ &= \sum_s d^{f,g}(s) \sum_{a^f} A_f^{f,g}(s, a^f) \nabla_\theta f(a^f|s). \end{aligned}$$

Note that the last equality holds because $\sum_{a^f} f(a|s) = 1$ implies that $\sum_{a^f} \nabla_\theta f(a^f|s) = 0$. Here $d^{f,g}(s)$ is the expected frequency of encountering state s discounted by γ. In other words, the performance of the agent is differentiable, and the current advantage function can be used to characterize the gradient. Particularly, under the tabular representation, a policy at each state s is

$$\boldsymbol{f}_s = (f_{s,1}, f_{s,2}, \ldots, f_{s,|\mathcal{A}_f|-1}, 1 - \sum_{i=1}^{|\mathcal{A}_f|-1} f_{s,i})$$

where the parameter θ for policy model f are variables $f_{s,1}, f_{s,2}, \ldots, f_{s,|\mathcal{A}_f|-1}$ at $s \in \mathcal{S}$. Therefore, for $i = 1, 2, \ldots, |\mathcal{A}_f|$,

$$\frac{\partial v^{f,g}(s_0)}{\partial f_{s,i}} = d^{f,g}(s) \Big(A_f^{f,g}\big(s, a^f(i)\big) - A_f^{f,g}\big(s, a^f(|\mathcal{A}_f|)\big) \Big).$$

Due to (1), we know that there is a positive probability to visit any $s \in \mathcal{S}$ when starting from s_0. Therefore,

$$d^{f,g}(s) > 0 \ \text{ for } s \in \mathcal{S}. \tag{2}$$

Based on optimality condition (3.17) in [2] when competing against a policy f, for optimal policy g^* satisfies

$$\begin{aligned} v^{f,g^*}(s) &= -\max_{a^g} Q_g^{f,g^*}(s, a^g), \\ Q_g^{f,g^*}(s, a^g) &= \sum_{s' \in \mathcal{S}, a^f \sim f(a|s)} p(s'|s, a^f, a^g) \Big[-R(s) + \gamma \max_{a'^g} Q_g^{f,g^*}(s', a'^g) \Big], \end{aligned}$$

for any $s \in \mathcal{S}$. Similarly, when playing against a policy g, optimal policy f^* satisfies

$$v^{f^*,g}(s) = \max_{a^f} Q_f^{f^*,g}(s, a^f),$$
$$Q_f^{f^*,g}(s, a^f) = \sum_{s' \in \mathcal{S}, a^g \sim g(a|s)} p(s'|s, a^f, a^g)\Big[R(s) + \gamma \max_{a'^f} Q_f^{f^*,g}(s', a'^f)\Big],$$

for any $s \in \mathcal{S}$.

The policy improvement theorem (inequalities (4.7) and (4.8) in [2]) when playing against a certain policy g, for two policies f and f', states that if

$$v^{f,g}(s) \leq Q_f^{f,g}(s, f'(a|s)) \triangleq \mathbb{E}_{a^f \sim f'(a|s)} Q_f^{f,g}(s, a^f)$$

holds for any state s, then

$$v^{f',g}(s) \geq v^{f,g}(s) \text{ for } s \in \mathcal{S}.$$

Similarly, if

$$v^{f,g}(s) \geq Q_g^{f,g}(s, g'(a|s)) \triangleq \mathbb{E}_{a^g \sim g'(a|s)} Q_g^{f,g}(s, a^g)$$

holds for any state s, then

$$v^{f,g'}(s) \leq v^{f,g}(s) \text{ for } s \in \mathcal{S}.$$

Although the policy improvement theorem was initially established for pure policies, the line of logic in its proof also holds for mixed policies.

Let us define set g_P to be the set containing any possible pure policy for agent g. Then it is clear that

$$\min_g v^{f,g}(s_0) = \min_{g \in g_P} v^{f,g}(s_0).$$

We now switch to the main part of the proof, which includes a few claims listed below.

For a given f, we consider the performance of g on the set g_P. We evaluate $v^{f,g}(s_0)$ for every $g \in g_P$, which yields the set $\left\{v^{f,g_{P_1}}(s_0), v^{f,g_{P_2}}(s_2), \ldots, v^{f,g_{P_K}}(s_0)\right\}$ of distinct values. We rank those values in the ascending order to get an ordered set

$$\left\{v^{(1)}(f),\ v^{(2)}(f),\ \ldots\right\}.$$

Here $v^{(1)}(f) = \min_{g \in g_P} v^{f,g}(s_0)$, $v^{(2)}(f)$ is the second smallest value, and so on. We have

$$\delta(f) = v^{(2)}(f) - v^{(1)}(f) > 0 \tag{3}$$

which is a strictly positive gap between the first and second smallest performance on set g_P. We also define the best response set

$$g_P^{\text{best}}(f) = \{g \in g_P \mid v^{f,g}(s_0) = v^{(1)}(f)\}.$$

Claim 1: For a policy vector $\boldsymbol{f} = (\boldsymbol{f}_{s_1}^T, \boldsymbol{f}_{s_2}^T, \ldots, \boldsymbol{f}_{s_{N_s}}^T)^T$ where $\boldsymbol{f}_{s_i} \in \mathbb{R}^{|\mathcal{A}_f|}$ is the policy vector at state s_i, a vector $\Delta_f = (\Delta_{f,1}, \Delta_{f,2}, \ldots, \Delta_{f,s_i})^T$ where $\Delta_{f,i} \in \mathbb{R}^{|\mathcal{A}_f|}$ and $\Delta_{f,i}^T \mathbf{1} = 0$ for $i = 1, 2, \ldots, N_s$, and an $\epsilon_1 > 0$ such that

$$\boldsymbol{f} + \epsilon_1 \Delta_f \geq 0, \tag{4}$$

we have

$$g_P^{\text{best}}(\boldsymbol{f} + \epsilon \Delta_f) \subseteq g_P^{\text{best}}(f)$$

for any $\epsilon > 0$ small enough.

Proof. Because of the policy gradient theorem and the fact that R is bounded due to the finite state space $\mathcal{S}$, there exists $M > 0$ such that

$$\Big|\frac{\partial v^{\tilde{f},\tilde{g}}(s_0)}{\partial \tilde{f}_{s,i}}\Big| < M \ \forall \tilde{f}, \tilde{g}, s \in \mathcal{S}, i = 1, 2, \dots, |\mathcal{A}_f| - 1.$$

By the assumption (4) of the claim, there exists a small enough $\epsilon_2 > 0$ such that for $0 < \epsilon < \min(\epsilon_1, \epsilon_2, \frac{\delta(f)}{2M})$, we have $\mathbf{0} \le \boldsymbol{f} + \epsilon\Delta_f \le \mathbf{1}$ (so it is still a well-defined policy) and, by the definition of the derivative and the definition of M, for any $g \in g_P$ we have

$$\left|v^{\boldsymbol{f}+\epsilon\Delta_f,g}(s_0) - v^{\boldsymbol{f},g}(s_0)\right| < \frac{\delta(f)}{2}.$$

Thus, for every $g \in g_P^{\text{best}}(f)$, since $v^{(1)}(f) = v^{f,g}(s_0)$, we have

$$v^{\boldsymbol{f}+\epsilon\Delta_f,g}(s_0) < v^{(1)}(f) + \frac{\delta(f)}{2}.$$

And for every $\tilde{g} \in g_P \setminus g_P^{\text{best}}(f)$, we have

$$v^{\boldsymbol{f}+\epsilon\Delta_f,\tilde{g}}(s_0) > v^{(2)}(f) - \frac{\delta(f)}{2} = v^{(1)}(f) + \frac{\delta(f)}{2}.$$

Therefore, for every $g \in g_P^{\text{best}}(f)$, we have $\tilde{g} \notin \operatorname{argmin}_g v^{\boldsymbol{f}+\epsilon\Delta_f,g}(s_0)$, which implies that given the feasible direction Δ_f for policy f with a corresponding $\epsilon_1 > 0$, for any small enough $\epsilon > 0$ we have

$$g_P^{\text{best}}(\boldsymbol{f} + \epsilon\Delta_f) \subseteq g_P^{\text{best}}(f). \tag{5}$$

This shows the claim. $\square$

Claim 2: Let f be a local maximum for F. Then for any $s \in \mathcal{S}$, the linear system

$$\begin{aligned} \Delta^T \boldsymbol{A}_s &> \mathbf{0} \\ \boldsymbol{f}_s + \Delta &\ge \mathbf{0} \\ (\boldsymbol{f}_s + \Delta)^T \mathbf{1} &= 1 \end{aligned} \tag{6}$$

is infeasible with $\Delta \in \mathbb{R}^{|\mathcal{A}_f|}$ as variables, where the advantage matrix is

$$\boldsymbol{A}_s = \Big(A^{f,g_{P_j}}\big(s, a^f(i)\big)\Big)_{i,j}$$

with $g_{P_j} \in g_P^{\text{best}}(f)$ for $j = 1, 2, \dots, |g_P^{\text{best}}(f)|$.

Proof. We show the statement by contradiction. Let us assume the existence of a state $s \in \mathcal{S}$ for which the linear system (6) is feasible with $\Delta = (\Delta_1, \Delta_2, \dots, \Delta_{|\mathcal{A}_f|})^T \in \mathbb{R}^{|\mathcal{A}_f|}$. This implies that the conditions of Claim 1 are met and thus $g_P^{\text{best}}(\boldsymbol{f} + \epsilon\Delta_f) \subseteq g_P^{\text{best}}(f)$ for any small enough $\epsilon > 0$. Here $\Delta_f = (\mathbf{0}^T, \mathbf{0}^T, \dots, \Delta^T, \dots, \mathbf{0}^T)^T$ with Δ being at the position corresponding to state s. Due to $\boldsymbol{f}_s^T\mathbf{1} = 1$, it is also obvious that $\Delta^T\mathbf{1} = 0$ and we have

$$\Delta_{|\mathcal{A}_f|} = -\sum_{i=1}^{|\mathcal{A}_f|-1} \Delta_i.$$

Observe that each row of $\boldsymbol{A}_s$ corresponds to an action in $\mathcal{A}_f$, and each column of $\boldsymbol{A}_s$ corresponds to a optimal pure policy for g when playing against the given f. This suggests that for $j = 1, 2, \dots, |g_P^{\text{best}}(f)|$ we have,

$$\sum_{i=1}^{|\mathcal{A}_f|-1} \Delta_i\Big(A^{f,g_{P_j}}\big(s, a^f(i)\big) - A^{f,g_{P_j}}\big(s, a^f(|\mathcal{A}_f|)\big)\Big) > 0.$$

By the policy gradient theorem, for $j = 1, 2, \dots, |g_P^{\text{best}}(f)|$ we have

$$\begin{aligned}
&\sum_{i=1}^{|\mathcal{A}_f|-1} \frac{\partial v^{f,g_{P_j}}(s_0)}{f_{s,i}} \Delta_i \\
&= d^{f,g_{P_j}}(s) \sum_{i=1}^{|\mathcal{A}_f|-1} \Delta_i \Big(A^{f,g_{P_j}}\left(s, a^f(i)\right) - A^{f,g_{P_j}}\left(s, a^f(|\mathcal{A}_f|)\right)\Big) \\
&> 0. \qquad (7)
\end{aligned}$$

The last inequality holds strictly since we have already argued that $d^{f,g}(s) > 0$ for every s, f, g when the starting point is the artificial state s_0. Inequality (7) shows that for any $g \in g_P^{\text{best}}(f)$, the directional gradient for v is strictly positive along Δ.

Therefore, there exists $\epsilon_3 > 0$ such that for $0 < \epsilon < \epsilon_3$ we have,

$$v^{\boldsymbol{f}_s+\epsilon\Delta, g_{P_j}}(s_0) > v^{f,g_{P_j}}(s_0) \text{ for } j = 1, 2, \dots, |g_P^{\text{best}}(f)|. \qquad (8)$$

Given Claim 1 and (8), we know that if at a certain state s there exists a vector Δ feasible to (6), then there exists $\tilde{\epsilon} > 0$ such that for any $0 < \epsilon < \tilde{\epsilon}$ we have,

$$\begin{aligned}
F(\boldsymbol{f}_s + \epsilon\Delta) &= \frac{1}{\gamma} \min_{g \in g_P} v^{\boldsymbol{f}_s+\epsilon\Delta, g}(s_0) \\
&= \frac{1}{\gamma} \min_{g \in g_P^{\text{best}}(\boldsymbol{f}_s+\epsilon\Delta)} v^{\boldsymbol{f}_s+\epsilon\Delta, g}(s_0) \\
&\geq \frac{1}{\gamma} \min_{g \in g_P^{\text{best}}(f)} v^{\boldsymbol{f}_s+\epsilon\Delta, g}(s_0) \quad \text{(due to (5))} \\
&> \frac{1}{\gamma} \min_{g \in g_P^{\text{best}}(f)} v^{f,g}(s_0) \quad \text{(due to (8))} \\
&= F(f).
\end{aligned}$$

We conclude that f can not be a local maximum of $F(f)$ as long as there exists a state s such that (6) is feasible. □

Claim 3: If f is a local maximum for F, then for every state s there exists a vector $\boldsymbol{w}_s$ such that

$$\Delta^T \boldsymbol{A}_s \boldsymbol{w}_s \leq 0, \boldsymbol{w}_s \geq 0, \boldsymbol{w}_s^T \boldsymbol{1} = 1,$$

for any vector Δ that makes $\boldsymbol{f}_s + \Delta$ a well-defined policy at state s.

Proof. First, we reorder the actions of f so that the one with the highest probability at state s is the last one in $\boldsymbol{f}_s$. Then, consider

$$\begin{aligned}
\tilde{\Delta}^T \tilde{\boldsymbol{A}}_s &> 0 \\
\tilde{\Delta}_i \geq 0 \ \ &\forall i \in \mathcal{C}_s
\end{aligned} \qquad (9)$$

where index set $\mathcal{C}_s = \{i \leq |\mathcal{A}_f| - 1 \mid f_{s,i} = 0\}$, vector $\tilde{\Delta} \in \mathbb{R}^{|\mathcal{A}_f|-1}$, and

$$\tilde{\boldsymbol{A}}_s = \left(\boldsymbol{A}_{s,1}^T - \boldsymbol{A}_{s,|\mathcal{A}_f|}^T, \boldsymbol{A}_{s,2}^T - \boldsymbol{A}_{s,|\mathcal{A}_f|}^T, \dots, \boldsymbol{A}_{s,|\mathcal{A}_f|-1}^T - \boldsymbol{A}_{s,|\mathcal{A}_f|}^T\right)^T$$

with $\boldsymbol{A}_{s,i}$ being the i-th row vector of $\boldsymbol{A}_s$.

By Claim 2, we know that (6) is infeasible. We now argue that if (6) is infeasible, so is (9). Let us assume that (9) has a solution $\tilde{\Delta}$. Then there exists a small enough $\epsilon > 0$ such that $f_{s,i} + \epsilon\tilde{\Delta}_i \geq 0$ for all $i = 1, 2, \dots, |\mathcal{A}_f| - 1$. This is true since $f_{s,i} = 0, \tilde{\Delta}_i \geq 0$ holds for every $i \in \mathcal{C}_s$, and for $i \notin \mathcal{C}_s$ we have $f_{s,i} > 0$ and thus the inequality holds for small enough $\epsilon > 0$. We denote one such appropriate value as ϵ_1. Finally, if we let $\Delta = (\tilde{\Delta}^T, -\tilde{\Delta}^T\mathbf{1})^T$, then we clearly have $\Delta^T\mathbf{1} = 0$, and $\boldsymbol{f}_s + \epsilon\Delta \geq \mathbf{0}$ for any ϵ with

$$0 < \epsilon < \epsilon_2 = \min\Big(\epsilon_1, \max(0, \frac{f_{s,|\mathcal{A}_f|}}{\tilde{\Delta}^T\mathbf{1}})\Big).$$

Since $f_{s,|\mathcal{A}_f|} > 0$, we have $\epsilon_2 > 0$. It is also easy to check that

$$\Delta^T\boldsymbol{A}_s = \epsilon\tilde{\Delta}^T\tilde{\boldsymbol{A}}_s > 0.$$

We therefore conclude that if f is a local maximum, then (9) is infeasible.

In (9), except for the strict inequality let us denote all other constraints as $\boldsymbol{B}\tilde{\Delta} \geq 0$. Note that $\boldsymbol{B}$ is disposed with 0,1 on the diagonal. By the theorem of alternatives, infeasibility of (9) implies that there exist $\boldsymbol{y} \geq 0, \boldsymbol{z} \geq 0$ so that

$$\tilde{\boldsymbol{A}}_s\boldsymbol{y} + \boldsymbol{B}^T\boldsymbol{z} = 0 \text{ and } \boldsymbol{y} \neq 0.$$

After rescaling $\boldsymbol{y}$, there exist $k > 0$ and column vector $\boldsymbol{w}_s \geq 0$ with $\boldsymbol{w}_s^T\mathbf{1} = 1$ such that

$$\tilde{\boldsymbol{A}}_s\boldsymbol{w}_s = -k\boldsymbol{B}^T\boldsymbol{z}.$$

Then, due to $\boldsymbol{B}\tilde{\Delta} \geq \mathbf{0}$ and $\boldsymbol{B}_{u,v} = 0$ for $u \notin \mathcal{C}_s, v \notin \mathcal{C}_s$, for any $\tilde{\Delta}$ with $\tilde{\Delta}_i \geq 0$ for $i \in \mathcal{C}_s$, we have

$$\tilde{\Delta}^T\tilde{\boldsymbol{A}}_s w_s = -k\tilde{\Delta}^T\boldsymbol{B}^T\boldsymbol{z} \leq 0. \tag{10}$$

We define a new vector Δ such that $\boldsymbol{f}_s + \Delta \geq \mathbf{0}, \Delta^T\mathbf{1} = 0$. Note that these are equivalent to saying that $\boldsymbol{f}_s + \Delta$ is a policy. We have $\Delta_i \geq 0$ for $i \in \mathcal{C}_s$ and $\Delta_{|\mathcal{A}_f|} = -\sum_{i=1}^{|\mathcal{A}_f|-1}\Delta_i$. If we let

$$\tilde{\Delta} = (\Delta_1, \Delta_2, \ldots, \Delta_{|\mathcal{A}_f|-1})^T,$$

then we have

$$\begin{aligned}
\tilde{\Delta}^T\tilde{\boldsymbol{A}}_s\boldsymbol{w}_s &= (\Delta_1, \Delta_2, \ldots, \Delta_{|\mathcal{A}_f|-1})\Big(\boldsymbol{A}_{s,1}^T - \boldsymbol{A}_{s,|\mathcal{A}_f|}^T, \boldsymbol{A}_{s,2}^T - \boldsymbol{A}_{s,|\mathcal{A}_f|}^T, \ldots, \boldsymbol{A}_{s,|\mathcal{A}_f|-1}^T - \boldsymbol{A}_{s,|\mathcal{A}_f|}^T\Big)^T\boldsymbol{w}_s \\
&= \Big(\sum_{i=1}^{|\mathcal{A}_f|-1}\Delta_i\boldsymbol{A}_{s,i} - \sum_{i=1}^{|\mathcal{A}_f|-1}\Delta_i\boldsymbol{A}_{s,|\mathcal{A}_f|}\Big)\boldsymbol{w}_s \\
&= (\Delta_1, \Delta_2, \ldots, \Delta_{|\mathcal{A}_f|-1}, -\sum_{i=1}^{|\mathcal{A}_f|-1}\Delta_i)\Big(\boldsymbol{A}_{s,1}^T, \boldsymbol{A}_{s,2}^T, \ldots, \boldsymbol{A}_{s,|\mathcal{A}_f|}^T\Big)^T\boldsymbol{w}_s \\
&= \Delta^T\boldsymbol{A}_s\boldsymbol{w}_s.
\end{aligned}$$

Together with (10), we thus have

$$\Delta^T\boldsymbol{A}_s\boldsymbol{w}_s \leq 0, \tag{11}$$

which shows the claim. □

Let f be a local maximum of F. Treating $\boldsymbol{w}_s$ defined in Claim 3 as coefficients for a convex combination of opponent's policies in set $g_P^{\text{best}}(f) = \{g_P^{f,1}, g_P^{f,2}, \ldots, g_P^{f,|g_P^{\text{best}}(f)|}\}$ (namely, the probability that the agent plays the $g_P^{f,j}$ at state s is equal to the j-th element of $\boldsymbol{w}_s$), we obtain a mixed policy $\tilde{g}_s^w$ at state s for the opponent agent g, the advantage function of which is $A_f^{f,\tilde{g}_s^w}\Big(s, a^f(i)\Big) = \big(\boldsymbol{A}_s\boldsymbol{w}_s\big)_i$

for action $a^f(i)$. Regarding this advantage function, we establish that

$$\begin{aligned}
v^{f,\tilde{g}_s^w}(s) &= \sum_{i=1}^{|\mathcal{A}_f|} f_{s,i}\ Q_f^{f,\tilde{g}_s^w}\Big(s, a^f(i)\Big) \text{ for } \ s \in \mathcal{S}, \text{ implies that} \\
0 &= \sum_{i=1}^{|\mathcal{A}_f|} f_{s,i}\bigg(Q_f^{f,\tilde{g}_s^w}\Big(s, a^f(i)\Big) - v^{f,\tilde{g}_s^w}(s)\bigg) \text{ for } s \in \mathcal{S}, \text{ and thus} \\
0 &= \sum_{i=1}^{|\mathcal{A}_f|} f_{s,i}\ A_f^{f,\tilde{g}_s^w}\Big(s, a^f(i)\Big) \text{ for } s \in \mathcal{S}, \text{ and} \\
0 &= \sum_{1 \le i \le |\mathcal{A}_f|, i \notin \mathcal{C}_s} f_{s,i}\ A_f^{f,\tilde{g}_s^w}\Big(s, a^f(i)\Big) \text{ for } s \in \mathcal{S}. \qquad (12)
\end{aligned}$$

Since $\boldsymbol{f}_s$ is a well-defined distribution, the set $\mathcal{C}_s^c = \{i \in \mathbb{N} \mid 1 \le i \le |\mathcal{A}_f|, f_{s,i} > 0\}$ is non-empty. With (12) and $\boldsymbol{f}_s \ge \mathbf{0}$, we see that there exist $j_1, j_2 \in \mathcal{C}_s^c$ (j_1 and j_2 can be identical) such that

$$A_f^{f,\tilde{g}_s^w}\Big(s, a^f(j_1)\Big) \le 0,\ f_{s,j_1} > 0, \qquad (13)$$

$$A_f^{f,\tilde{g}_s^w}\Big(s, a^f(j_2)\Big) \ge 0,\ f_{s,j_2} > 0. \qquad (14)$$

Claim 4: We have

$$A_f^{f,\tilde{g}_s^w}\Big(s, a^f(i)\Big) \le 0 \text{ for } \forall\ a^f(i). \qquad (15)$$

Proof. If (15) does not hold, then there exists an index i_1 such that $A_f^{f,\tilde{g}_s^w}\Big(s, a^f(i_1)\Big) = \big(\boldsymbol{A}_s \boldsymbol{w}_s\big)_{i_1} > 0$. Let Δ^1 be the vector defined as

$$\Delta_l^1 = \begin{cases} 1 & \text{if } l = i_1 \\ -1 & \text{if } l = j_1 \\ 0 & \text{otherwise.} \end{cases}$$

(Note that $f_{s,i_1} < 1$, since otherwise, $\mathcal{C}_s^c$ contains only i_1, and (12) implies $A_f^{f,\tilde{g}_s^w}\Big(s, a^f(i_1)\Big) = 0.$)

Meanwhile, given (13) we have $f_{s,j_1} > 0$. Therefore, there exists a $0 < \epsilon_{(1)} < \min(1 - f_{s,i_1},\ f_{s,j_1})$ such that

$$\begin{aligned}
\mathbf{0} &\le \boldsymbol{f}_s + \epsilon_{(1)}\Delta^1 \\
&= \Big(f_{s,1}, f_{s,2}, \ldots, f_{s,i_1} + \epsilon_{(1)}, \ldots, f_{s,j_1} - \epsilon_{(1)}, \ldots, f_{s,|\mathcal{A}_f|}\Big) \\
&\le \mathbf{1},
\end{aligned}$$

and

$$\epsilon_{(1)}(\Delta^1)^T \boldsymbol{A}_s \boldsymbol{w}_s = \epsilon_{(1)}\bigg(A_f^{f,\tilde{g}_s^w}\Big(s, a^f(i_1)\Big) - A_f^{f,\tilde{g}_s^w}\Big(s, a^f(j_1)\Big)\bigg) > 0,$$

which contradicts Claim 3. Therefore, such i_1 does not exist, and advantage for any action $a^f(i)$ is non-positive. □

Claim 5: We have

$$A_f^{f,\tilde{g}_s^w}\Big(s, a^f(i)\Big) = 0 \text{ for } \forall\ a^f(i) \text{ with } f_{s,i} > 0. \qquad (16)$$

Proof. If (16) does not hold, then there exists an index i_2 such that $A_f^{f,\tilde{g}_s^w}\left(s, a^f(i_2)\right) = \left(\boldsymbol{A}_s \boldsymbol{w}_s\right)_{i_2} < 0$ and $f_{s,i_2} > 0$. Let Δ^2 be the vector defined as

$$\Delta_l^2 = \begin{cases} -1 & \text{if } l = i_2 \\ 1 & \text{if } l = j_2 \\ 0 & \text{otherwise.} \end{cases}$$

(Note that $f_{s,j_2} < 1$ because we already have $f_{s,i_2} > 0$.) Therefore, there exists a $0 < \epsilon_{(2)} < \min(1 - f_{s,j_2},\ f_{s,i_2})$ such that

$$\begin{aligned} \mathbf{0} &\leq \boldsymbol{f}_s + \epsilon_{(2)} \Delta^2 \\ &= \left(f_{s,1}, f_{s,2}, \ldots, f_{s,i_2} - \epsilon_{(2)}, \ldots, f_{s,j_2} + \epsilon_{(2)}, \ldots, f_{s,|\mathcal{A}_f|}\right) \\ &\leq \mathbf{1}, \end{aligned}$$

and

$$\epsilon_{(2)} (\Delta^2)^T \boldsymbol{A}_s \boldsymbol{w}_s = \epsilon_{(2)} \left(A_f^{f,\tilde{g}_s^w}\left(s, a^f(j_2)\right) - A_f^{f,\tilde{g}_s^w}\left(s, a^f(i_2)\right) \right) > 0,$$

which contradicts Claim 3. Therefore, such i_2 does not exist, and advantage is zero for any action $a^f(i)$ with $f_{s,i} > 0$. □

In summary, the claims show that when policy f is a local maximum, we obtain a mixed policy $\tilde{g}_s^w$ for any state s. We denote the entire policy function as $\tilde{g}^w$. Obviously, $\tilde{g}^w$ only plays optimal actions against f and is also a best response to f. Meanwhile, under this policy $\tilde{g}^w$, due to (15) and (16), at any state s we have

$$A_f^{f,\tilde{g}_s^w}\left(s, a^f(i)\right) \leq 0 \text{ for } i = 1, 2, \ldots, |\mathcal{A}_f|,$$
$$A_f^{f,\tilde{g}_s^w}\left(s, a^f(i)\right) = 0 \text{ if } f_{s,i} > 0.$$

Therefore, f is also the best response to $\tilde{g}^w$ as prescribed by the Bellman optimality condition

$$v^{f,\tilde{g}^w}(s) = \max_{a^f \in \mathcal{A}_f} Q_f^{f,\tilde{g}^w}(s, a^f) \quad s \in \mathcal{S}.$$

We use this to show that f is the global maximum of function F. Since f is the best response to $\tilde{g}^w$, for any policy $\tilde{f}$, under $\tilde{g}^w$ we see that

$$\begin{aligned} v^{f,\tilde{g}^w}(s) &= \max_{a^f \in \mathcal{A}_f} Q_f^{f,\tilde{g}^w}(s, a^f) \\ &\geq \mathbb{E}_{a^f \sim \tilde{f}(a|s)} Q_f^{f,\tilde{g}^w}(s, a^f) \\ &= Q_f^{f,\tilde{g}^w}(s, \tilde{f}(a|s)) \end{aligned}$$

holds for any state s. By the policy improvement theorem, we thus have

$$v^{f,\tilde{g}^w}(s) \geq v^{\tilde{f},\tilde{g}^w}(s) \quad \text{for } \forall s.$$

Since $\tilde{g}^w$ is a best response to f, we know that

$$F(f) = \frac{1}{N_s} \sum_{s \in \mathcal{S}} v^{\tilde{f},\tilde{g}^w}(s).$$

Therefore,

$$F(\tilde{f}) \leq \frac{1}{N_s} \sum_{s \in \mathcal{S}} v^{\tilde{f},\tilde{g}^w}(s) \leq \frac{1}{N_s} \sum_{s \in \mathcal{S}} v^{f,\tilde{g}^w}(s) = F(f).$$

This concludes the proof. □

A.4 Demonstrations of the recovered IRL policies

Performances of the algorithm using $\mathcal{D}_{\epsilon=.1}$ are shown in Fig. 1. First of all, Fig. 1(a) shows that the IRL loss function (5) is improving during training. By IRL loss we refer to $\hat{v}^f - \hat{v}^g$ based on definitions in steps 9 and 10 in Algorithm 1, namely the objective function (5) of our IRL algorithm (without the regularization term $\phi(\theta_R)$). This trend suggests that $R_{\theta_R}(s)$ gradually learns to explain the behaviors in the demonstration set. Meanwhile, as discussed above, the success of the IRL algorithm relies on the quality of the Nash Equilibrium policy models we maintain during IRL training. Although θ_R is being updated continuously and the Nash Equilibrium polices are expected to be changing during training, Fig. 1(b) shows that the gaps between the performances of $f_{\theta_f}, g_{\theta_g}$ and their best possible performances are pretty marginal, thus indicating the good quality of both policy models. The plot depicts $v^{f_{\theta_f}, g_{\theta_g}}(s_0; R_{\theta_R}), \min_g v^{f_{\theta_f}, g}(s_0; R_{\theta_R}), \max_f v^{f, g_{\theta_g}}(s_0; R_{\theta_R})$, and shows that the three values are close to each other for most of the iterations during training. Regarding the property of the obtained reward function, Fig. 1(c) reveals a strong correlation ($\rho = 0.65, p < .001$) between $R_{\text{chasing}}(s)$ and the $R_{\theta_R}(s)$ we recovered after 500,000 iterations of training. This strong correlation indicates that the model learns that the reward of each state should be highly dependent on $D(s)$ and behaves similarly as $R_{\text{chasing}}(s)$.

To further corroborate the quality of the recovered reward and policy functions we include two more metrics. First, we compare the divergence between the IRL and Nash Equilibrium policies. As shown in Fig. 1(d), we gauge the KL-divergence between the IRL and Nash Equilibrium policies and plot the estimation performed on a batch of 64 randomly sampled states. When compared against a model that acts randomly or the “early” policy models obtained after 20,000 iterations, the IRL policies demonstrate behaviors that are most similar to those of the Nash Equilibrium policies.

A more direct measurement is to plug IRL policies back into the chasing game and evaluate their performances when competing against the Nash Equilibrium policies. In Fig. 1(e) we depict the performances of the IRL and Nash Equilibrium polices estimated in 64 rounds of games. The policies $f_{\theta_f}, g_{\theta_g}$ obtained after 500,000 iterations of IRL training demonstrate performances that are relatively close to those of Nash Equilibrium policies.

In Fig. 2 we demonstrate the behaviors of IRL policies based on random trajectories generated by $f_{\theta_f}, g_{\theta_g}$. Despite occasional mistakes (for example, in the first row of Fig. 2 one of the predators chose not to move), the two predators are pursuing the preys in a coordinated way, and the preys are actively keeping a distance from the predators.

There remains a concern on whether the prior knowledge in our regularization function is too strong. If so, we should have approximated $R_{\text{chasing}}(s)$ decently well from the beginning of our training. We clear this doubt by inspecting the Nash Equilibrium policies early in the algorithm. In Fig. 3, we show trajectories generated by policy models at the 20,000-th iteration. Note that we also use models obtained at the 20,000-th iteration as the “early” models in Fig. 1 because at the 20,000-th iteration the Nash Equilibrium polices are of good qualities already (shown in Fig. 1(b)) while IRL training has just begun (shown in Fig. 1(a)). Obviously, for trajectories in Fig. 3 both agents act remarkably different from policies shown in Fig. 2. To be specific, in Fig. 3 the preys try to move to and stay at two corners on the diagonal of the grid, while the predators try to stay on the diagonal of the two preys. This is not surprising since in $\phi(\theta_R)$ we are encouraging the average distance $\bar{D}(s)$ and $R(s)$ to be correlated instead of the max-min distance $D(s)$ and $R(s)$, and the behaviors of predators/preys serve to minimize/maximize $\bar{D}(s)$. Therefore, we confidently draw the conclusion that the policies and the reward function are recovered by our IRL algorithm because of the correctly proposed objective function that minimizes the performance gap rather than the prior knowledge provided by the regularization terms.

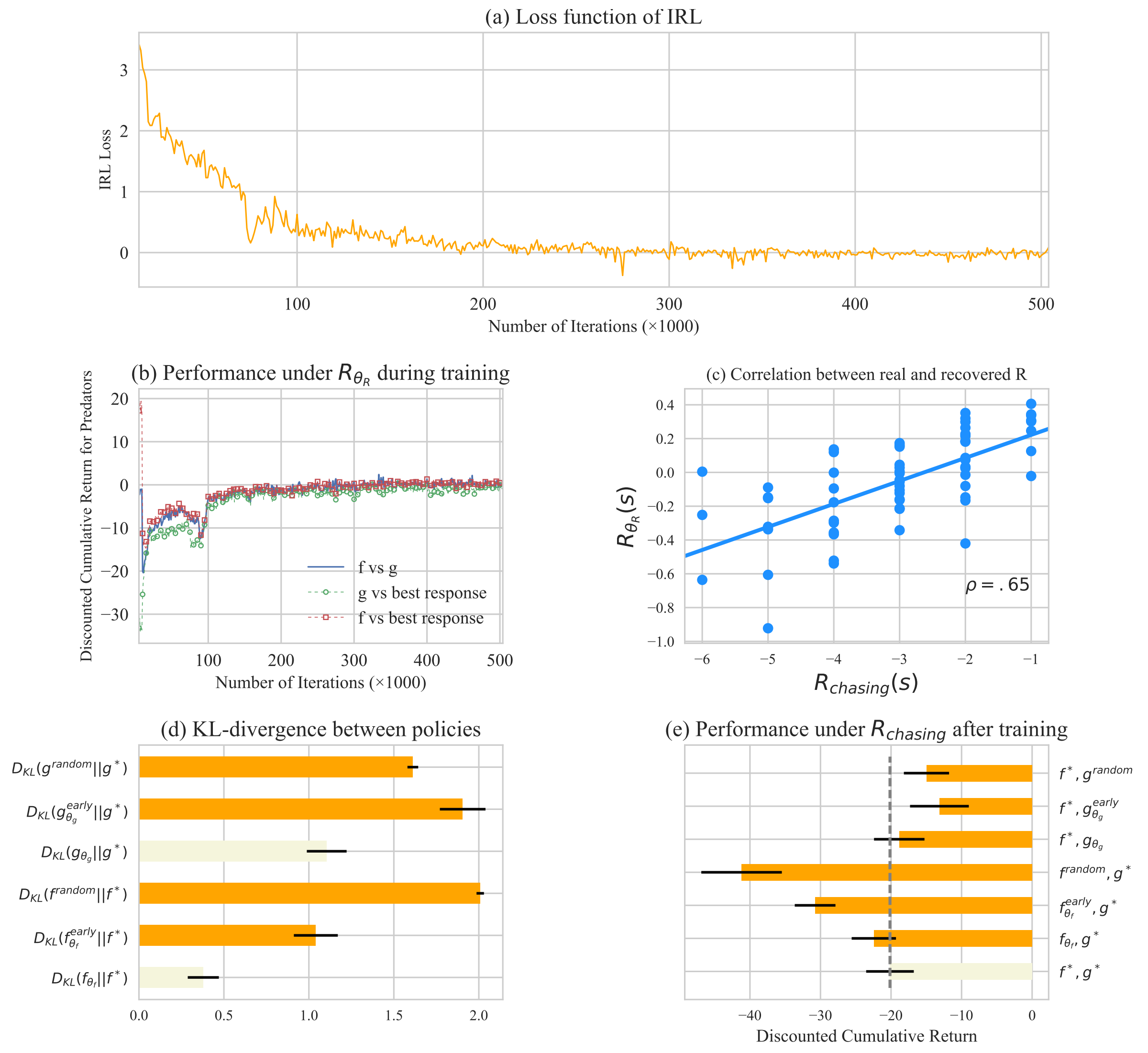

Figure 1: Results of IRL training. (a) The IRL loss function, which is an estimation of objective function based on sampled trajectories, is decreasing during training. This trend indicates that R_{θ_R} gradually learns to explain the expert demonstrations in our training. (b) Performances of Nash Equilibrium policy models and their best response models during training. For the majority of the iterations, the gap of performances between $f_{\theta_f}, g_{\theta_g}$ and the best response opponent models is marginal, suggesting that $f_{\theta_f}, g_{\theta_g}$ are close enough to Nash Equilibrium policies $f^*(R_{\theta_R}), g^*(R_{\theta_R})$ during IRL training. (c) The recovered reward function $R_{\theta_R}(s)$ demonstrates a strong correlation to $R_{\text{chasing}}(s)$ ($p < .001$). The two scales are different as reward functions are identical up to a scaling factor. (d) KL-divergence between f_{θ_f}(or g_{θ_g}) and $f^*(R_{\text{chasing}})$(or $g^*(R_{\text{chasing}})$). "Early" denotes the models at the 20,000-th iteration, while "random" model follows a uniform distribution on all the 5 available actions. The final results of IRL training are as expected most similar to Nash Equilibrium policies. Error bars indicate the standard errors estimated on a batch of 64 samples. (e) Performance of policy models under R_{chasing}. The dashed reference line represents the performance of the Nash Equilibrium model. Policies recovered by IRL training play similarly well when compared with the Nash Equilibrium policy, while "early" and "random" models exhibit much more significant performance gaps. Error bars indicate the standard deviation estimated on a batch of 64 samples.

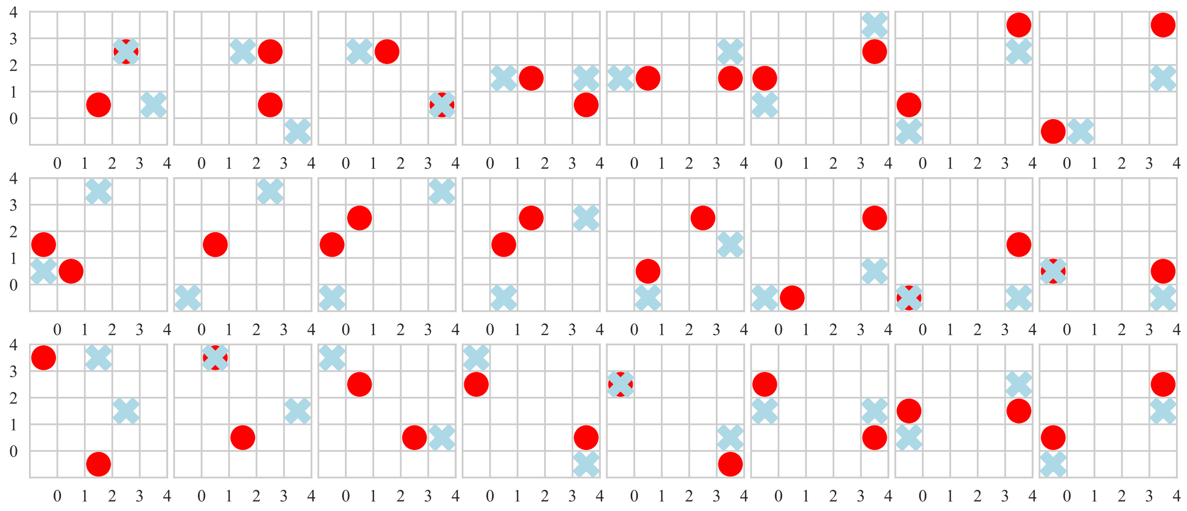

Figure 2: Trajectories generated by policy models obtained in the IRL algorithm. We use red dots to represent predators and blue crosses for preys. Each row presents a different 8-step trajectory.

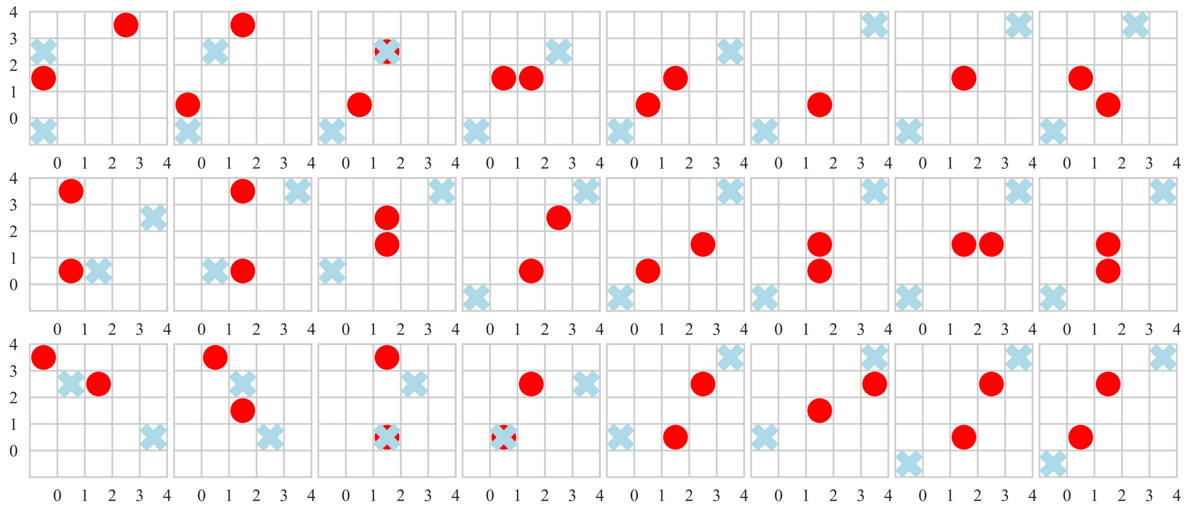

Figure 3: Trajectories generated by policy models obtained at the 20,000-th iteration. Each row presents a different 8-step trajectory. Clearly, these actions are not driven by $R_{\text{chasing}}(s) = D(s)$ and are different from the ones in Fig. 2.

A.5 Implementations and Performances of Benchmark IRL Algorithms

Our IRL algorithm demonstrated is designed and tailored for two specific goals: to take sub-optimality of expert demonstrations into account, and to cope with zero-sum games of large scales. We next illustrate the superior performance of our algorithm in the chasing game regardless of the quality of the demonstration set. The Bayesian-IRL [3] (BIRL) algorithm and Decentralized-IRL [4] (DIRL) are selected as benchmark IRL algorithms since they are the only ones that solve competitive multi-agent IRL tasks to the best of our knowledge. Both algorithms need modifications since deep neural nets should be used as model approximations and the IRL training should proceed efficiently for large-scale games that cannot be solved by tabular approaches with enumeration of states or actions. We next provide the details.

The BIRL algorithm for zero-sum stochastic games formulates a quadratic programming problem with constraints that require each demonstrated action to be the optimal one, and the objective function represents the posterior of the current reward function given a Bayesian prior of reasonable reward functions. Two problems arise when implementing the BIRL algorithm to solve the chasing game. First, enumerating and imposing all the constraints is not tractable (there are $2 \cdot N_s \cdot |\mathcal{A}| = 19,531,250$ constraints in the current version of the game). Therefore, for each iteration in our training, we sample a tuple $(s, a^{E,f}, a^{E,g})$ from $\mathcal{D}$, and randomly choose actions (a^f, a^g) from $\mathcal{A}^f \times \mathcal{A}^g$. In an iteration we consider only the constraints on the sampled state-action pairs. By adding Lagrangians into the objective function to encourage the constraints, we charge a penalty whenever the demonstrated actions $a^{E,f}, a^{E,g}$ are performing worse than a^f, a^g.

Another issue is that the BIRL algorithm requires explicit models for expert policies. The aforementioned constraints in the BIRL algorithm are equivalent to $Q_f^{f^E,g^E}(s, a^{E,f}) \geq Q_f^{f^E,g^E}(s, a^f)$ and $Q_g^{f^E,g^E}(s, a^{E,g}) \geq Q_g^{f^E,g^E}(s, a^g)$. Evaluation of the Q-functions are feasible only if f^E, g^E and the corresponding state transition matrix are available and can be stored in a computer's memory. In [3] the expert policies are statistically recovered since sufficient demonstrations are available for a significantly smaller game, which is impossible for large games. Instead of appealing to imitation learning to yield expert model approximations, we conduct a two-phase training that does not rely on expert policy models. We find the optimal state value function first, and then use the state-value function and one-step transition probability (which can be approximated by sampling tuples from $\mathcal{D}$) to recover R. To be specific, the BIRL algorithm is based on the equality $\boldsymbol{V} = (\boldsymbol{I} - \gamma \boldsymbol{P})^{-1} \boldsymbol{R}$ where $\boldsymbol{V} = \big(V(s)\big)_{s \in \mathcal{S}}$ is the vector exhibiting the value for each state, $\boldsymbol{R} = \big(R(s)\big)_{s \in \mathcal{S}}$ is the vector for the reward at each state, and $\boldsymbol{P}$ is the state transition matrix under expert policies. To infer $\boldsymbol{V}$ from $\boldsymbol{R}$, the inversion of $(\boldsymbol{I} - \gamma \boldsymbol{P})^{-1}$ necessitates expert policy models that can act in all states (including those not demonstrated in $\mathcal{D}$) and generate infinitely long trajectories. Instead, if the algorithm first finds state value functions $V(s)$ instead of $R(s)$ and uses $\boldsymbol{R} = (\boldsymbol{I} - \gamma \boldsymbol{P}) \boldsymbol{V}$ to recover R, then only one-step transitions that can be sampled directly from $\mathcal{D}$ are needed. Therefore, in our implementation the first phase of training uses the BIRL algorithm to solve for $v_{\theta_V}^{f^E,g^E}(s)$, the vector representation of which is the vector $\boldsymbol{V}$ above. The objective function is equal to the Lagrangian terms plus the same regularization term $\phi(\theta_R)$ used for our IRL algorithm (now viewed as prior of R in BIRL). In the second phase, we sample a state s and corresponding expert actions from $\mathcal{D}$, get the following state s' under the known transition function, and train θ_R to minimize the squared loss between $R_{\theta_R}(s)$ and $v_{\theta_V}^{f^E,g^E}(s) - \gamma v_{\theta_V}^{f^E,g^E}(s')$. Note that the objective function in the R-phase is also regularized by the same $\phi(\theta_R)$ in the V-phase, because in our experiments we have observed a drastic deterioration of performances if the regularization term is not used for both phases. The model $v_{\theta_V}^{f^E,g^E}(s)$ and reward $R_{\theta_R}(s)$ are parametrized similarly as specified in Section 4.2. Lastly, since the BIRL algorithm returns only a reward function, we use the proposed Nash Equilibrium algorithm to solve for $f^*(R_{\theta_R}), g^*(R_{\theta_R})$ after two-phase training.

The DIRL algorithm also assumes the optimality of expert policies under the unknown reward function. The algorithm alternates between a π step and an R step. In the k-th iteration of training, the algorithm first enters the π step that solves for the Nash Equilibrium policies (f_k, g_k) under current R_{θ_R}. The policies (f_k, g_k) are added into a policy set Π. Then in the R step, the algorithm finds R_{θ_R} that maximizes $\frac{1}{k} \sum_{j=1}^{k} \sum_{s \in \mathcal{S}} p\Big(v^{f^E,g^E}(s) - v^{f_j,g^E}(s)\Big) + p\Big(v^{f^E,g_j}(s) - v^{f^E,g^E}(s)\Big) - \phi(\theta_R)$, where $p(x) = \max(x, 0) + 2 \cdot \min(x, 0)$. This objective function encourages R to favor f^E, g^E when competing against any policies in Π, which is aligned with the optimality assumption that (f^E, g^E) are indeed Nash Equilibrium of the game.

To alleviate the overhead of storing and calling all the policies in Π, in the R step of our deep implementation we maximize a slightly different objective function $\mathbb{E}_{(f_j,g_j) \sim \Pi} \mathbb{E}_{(s,\_,\_) \sim \mathcal{D}} \Big[p\Big(v^{f^E,g^E}(s) -$

Table 1: Correlations between recovered reward functions and R_{chasing}

IRL Algorithm	$\epsilon = .05$	$\epsilon = .1$	$\epsilon = .2$
Algorithm 1	0.65	0.68	0.66
BIRL	0.28	-0.02	0.12
DIRL	-0.31	-0.15	0.11

Table 2: Performance deterioration of recovered policies under R_{chasing} (*:Nash Eq.; A:Algorithm 1; B:BIRL; D:DIRL)

$\mathcal{D}_\epsilon$	f^{A}	f^{B}	f^{D}	g^{A}	g^{B}	g^{D}
$\epsilon = .05$	11.8%	24.1%	197.0%	4.4%	33.5%	38.9%
$\epsilon = .1$	10.3%	44.3%	100.0%	6.9%	33.5%	41.9%
$\epsilon = .2$	13.3%	68.5%	200.1%	9.3%	33.0%	40.9%

$v^{f_j,g^E}(s)\Big) + p\Big(v^{f^E,g_j}(s) - v^{f^E,g^E}(s)\Big) - \phi(\theta_R)\Big]$. Thus, in each training iteration we only sample one pair of (f_j, g_j) from Π and pit them against expert policies, so the expectation of this new objective function remains unchanged when compared with the original one. Again, models of f^E, g^E are still required to evaluate the state value functions. Here we adopt the treatment of the original DIRL work [4] and our IRL algorithm; we let f^E, g^E to act at the first step of each trajectory by sampling from $\mathcal{D}$, then we use the latest (f_k, g_k) to act for all the following steps to generate the full trajectory.

To make sure we are performing a fair comparison, the number of training iteration for each algorithm is set to match the runtime of all 3 algorithms. For both phases in the algorithm, training lasts for 500,000 iterations. For the Lagragians in BIRL training, we use a fixed coefficient for all constraints instead of a unique and dynamically updated coefficient for each one, which would theoretically require another neural network models for $\lambda(s, a)$. Besides, the fixed coefficient is set to be 1 since we can change the weights in $\phi(\theta_R)$ instead, which is similar to the original treatment in [3]. We set the weight coefficient c in $\phi(\theta_R)$ to be .25 to match up with the one in our algorithm. The DIRL algorithm is computationally demanding largely due to the time spent on solving for Nash Equilibrium at each iteration of training. To control the runtime of the DIRL algorithm within a comparable range of the other IRL methodologies in our experiment, we perform 10 iterations with 50,000 training iterations for both the π and R steps in each iteration. For both algorithms, policies and reward function models are parametrized similarly as specified in Section 4.2. The learning rate is set to be 10^{-4}, weight c of regularization term is set as .25 and Adam [5] is used as optimizer. We mention that, even under such a specification, DIRL training still more than tripled the runtime of the other algorithms in our experiments.

We summarize the result of experiments in Table 1 and 2. Under the same regularization terms, only our IRL algorithm finds a reward function that bears reasonably high correlation with $R_{\text{chasing}}(s)$. We also plug the solved IRL policies back into the original chasing game and compete against the Nash Equilibrium policies, and measure the gap between their performances and v^{f^*,g^*} to evaluate how much the performance of the recovered policies deteriorate. As shown in Table 2, only our IRL algorithm recovers policies of good quality, and the performance is not largely affected by the demonstration set we use. The issues with the BIRL algorithm are the requirement of accurate expert policy models and the strict optimality constraints of expert actions, whereas the number of reference policies in Π is likely to be highly critical to the success of the DIRL algorithm. In conclusion, our IRL algorithm overcomes the issues in the benchmark algorithms, and outperforms them significantly when all the algorithms are implemented and utilized in the same setting.

A.6 Demonstrations of Solved Nash Equilibrium policies

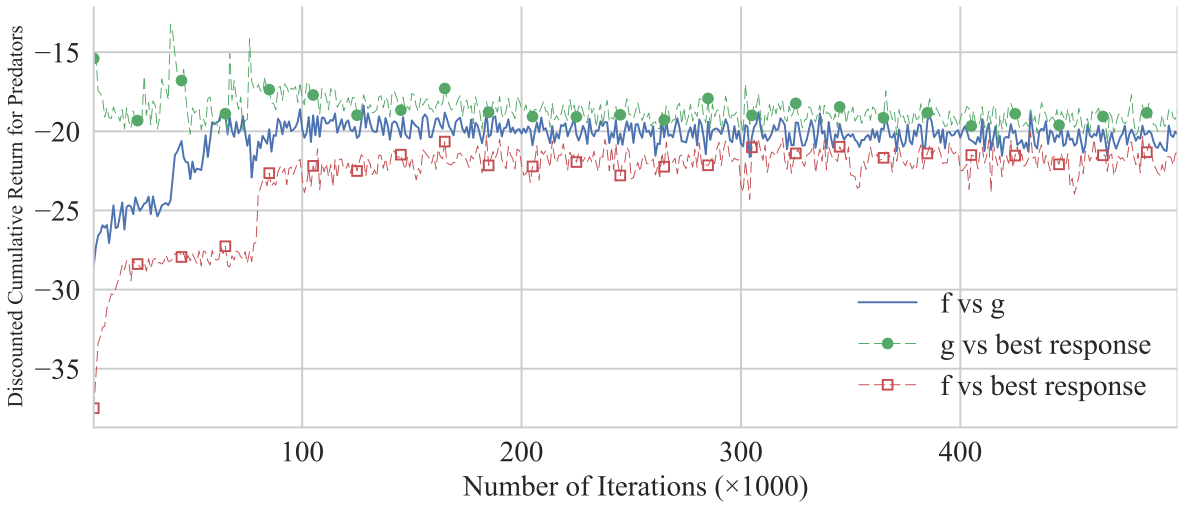

Figure 4: Performance of the proposed Nash Equilibrium algorithm in the chasing game

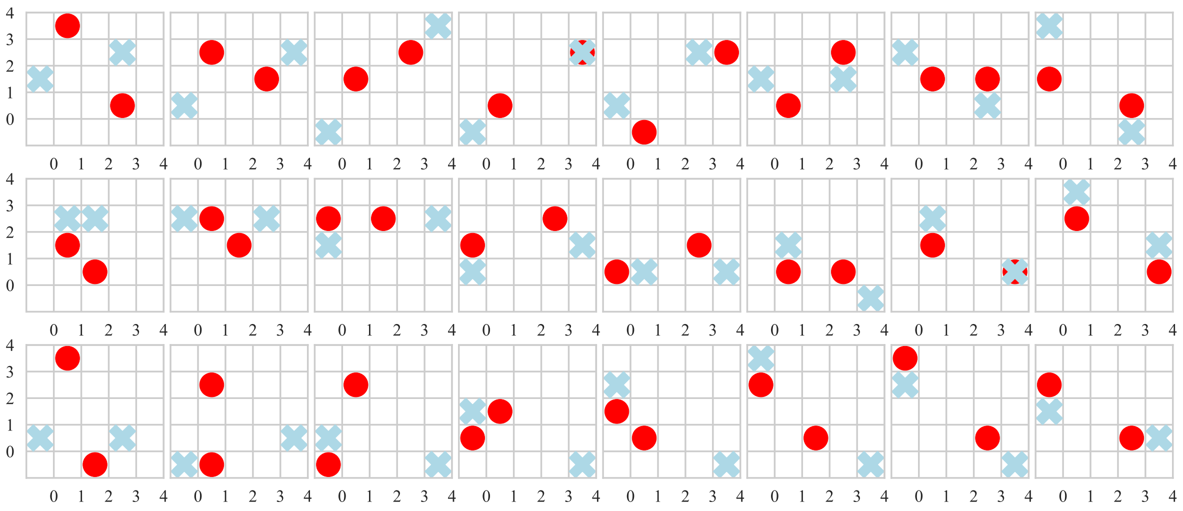

Figure 5: Trajectories generated by the trained models in the chasing game. Each row presents a different 8-step trajectory.

Fig. 4 shows the evolution of the performances of policy models. The plot depicts $v^{f_{\theta_f}, g_{\theta_g}}(s_0; R_{\text{chasing}})$, $\min_g v^{f_{\theta_f}, g}(s_0; R_{\text{chasing}})$, $\max_f v^{f, g_{\theta_g}}(s_0; R_{\text{chasing}})$. Recall that if f_{θ_f} and g_{θ_g} manage to reach the Nash Equilibrium, the three series should converge perfectly. As the figure shows, the adversarial training on f_{θ_f} and g_{θ_g} succeed in decreasing the gap to a pretty marginal level, suggesting that f_{θ_f} and g_{θ_g} should be close enough to the Nash Equilibrium policies and there is little room to further improve their performances.

To further corroborate the quality of the learned policies, in Fig. 5 we plot trajectories showing behaviors of the obtained f_{θ_f} and g_{θ_g}. As demonstrated in the figure, both the predators and the preys understand their goals in the game. The predators allocate the tasks so that they are not chasing the same prey and ignoring the other one, while the policy of the preys is that they run away actively and try not to stay at the same cell, thus making it harder for the predators to pursue both of them.

A.7 Implementation and Performance of Benchmark Nash Equilibrium Algorithm

To further demonstrate the superiority of the proposed Nash Equilibrium algorithm particularly for large games, we reformulate the quadratic programming problem proposed on page 125 in [6], which inspires the gradient descent algorithm to solve for Nash Equilibrium proposed in [7]. We select only this algorithm as the benchmark in this section, since the algorithm in [8], due to its formulation, was observed to provide a zero gradient to policies when training in zero-sum games, and we do not find an easy solution to apply the algorithm in [9] to large games using deep neural nets as model approximations.

Here we illustrate the deep implementation we used for the benchmark algorithm in the experiment. The algorithm maintains both policy models f, g and models for bounds of state value functions $v^f(s), v^g(s)$. The softmax layers in policy models guarantee $f(a|s), g(a|s)$ to be well-defined distributions. Therefore, the remaining constraints are

$$R(s) + \gamma \mathbb{E}_{a^g \sim g(a|s), s' \sim p(s'|s,a^f,a^g)} v^f(s') \leq v^f(s) \text{ for any } s, a^f,$$

$$-R(s) + \gamma \mathbb{E}_{a^f \sim f(a|s), s' \sim p(s'|s,a^f,a^g)} v^g(s') \leq v^g(s) \text{ for any } s, a^g,$$

and the objective is to minimize $\mathbb{E}_s[v^f(s) + v^g(s)]$. The constraints are implemented as Lagrangians with a coefficient λ. Similarly to the implementation of the benchmark BIRL algorithm, we do not enumerate all the constraints, but only sample s from S and a^f, a^g from $f(a|s), g(a|s)$ at each iteration of training. Whenever a constraint on the sampled state-action pair is violated, a penalty is added to the objective function that motivates f, g to avoid taking sub-optimal actions. To evaluate the expectation in each constraint, we sample 5 trajectories for each constraint. Training lasts for 500,000 iterations. Both policies and value models are parametrized as in our Nash Equilibrium algorithm. Theoretically speaking, for each state-action pair, the corresponding constraint should have its own λ (which necessitates an extra neural network model) that would be constantly updated in each iteration. Since we are not enumerating the constraints, we set λ to be a fixed value throughout training, and conduct grid search on the optimal value of λ. According to our experiments, performance of the algorithm is not very sensitive to the value λ, and we use $\lambda = 10$ for the results shown in this section since it appears to be the optimal value in our experiments. The results are shown in Table 3 of Section 4.3.

References

[1] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[2] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[3] Xiaomin Lin, Peter A Beling, and Randy Cogill. Multi-agent inverse reinforcement learning for two-person zero-sum games. *IEEE Transactions on Computational Intelligence and AI in Games*, 2017.

[4] Tummalapalli Reddy, Vamsikrishna Gopikrishna, Gergely Zaruba, and Manfred Huber. Inverse reinforcement learning for decentralized non-cooperative multiagent systems. In *Systems, Man, and Cybernetics*, pages 1930–1935, 2012.

[5] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[6] Jerzy Filar and Koos Vrieze. *Competitive Markov decision processes.* Springer Science & Business Media, 2012.

[7] Prasad H.L. and Shalabh Bhatnagar. A study of gradient descent schemes for general-sum stochastic games. *arXiv preprint arXiv:1507.00093*, 2015.

[8] Prasad H.L., Prashanth L.A., and Shalabh Bhatnagar. Two-timescale algorithms for learning nash equilibria in general-sum stochastic games. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pages 1371–1379, 2015.

[9] Natalia Akchurina. Multiagent reinforcement learning: algorithm converging to nash equilibrium in general-sum discounted stochastic games. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 725–732, 2009.