EARL: THE ART OF PEER PRESSURE

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-agent reinforcement learning (MARL) has shown great success in zero-sum games. Yet, in general-sum games, the interaction of self-interested agents regularly leads to collectively worst outcomes. To overcome this, methods such as Learning with Opponent Learning Awareness (LOLA) and Model-Free Opponent Shaping (M-FOS) shape their co-player's learning. Shaping co-players has proven successful, leading often to more pro-social outcomes in general-sum games. However, previous shaping methods are either: *myopic* – only shape the co-player's next few learning steps; require *white-box access* to the parameters and gradients of the co-player; or *do not scale* with the size of deep neural networks use to parameterized. To address these issues, we propose EARL. A EARL agent is a meta-learner parameterized by a recurrent neural network, that meta-learns to shape their co-player over multiple trials. EARL is model-free, considers long time horizons, successfully scales to policies parameterized by deep neural networks. In a set of experiments, we show that EARL achieves optimal behavior on matrix games in line with existing methods with no privileged information and achieves state-of-the-art performance on grid-world based games. EARL shapes co-players in high dimensional environments where previous approaches fail. Furthermore, we propose revisions to existing environments and a new experimental protocol to identify and disentangle degenerate training cases from opponent shaping.

1 INTRODUCTION

Many real-world scenarios, i.e. self-driving cars or diplomacy, contain both competitive and cooperative elements and are thus ideally modelled as mixed-motive games. The players are constantly learning and affect each others learning dynamics, rendering the scenarios non-stationary. It is the goal of multi-agent learning to develop methods which can operate optimally and robustly in such scenarios.

Multi-agent learning has shown great success in strictly competitive (capture-the-flag (Jaderberg et al., 2019)) and cooperative scenarios (Overcooked (Strouse et al., 2021)). In such games, a best-response to fixed co-players leads to the Nash Equilibrium. In contrast, in mixed-motive games, this strategy can lead to the globally worst outcome. For example, in the iterative prisoner's dilemma (IPD) (Axelrod & Hamilton, 1981; Harper et al., 2017), learning agents that do not account for other agents' learning converge on unconditional mutual defection.

Shaping methods explicitly account for other agent's learning and have achieved promising results, such as finding the prosocial tit-for-tat strategy in the IPD (Foerster et al., 2018a; Letcher et al., 2019b; Willi et al., 2022; Balaguer et al., 2022; Lu et al., 2022b). However, early shaping methods were myopic (they only shape the next learning step of the co-player), required white-box access to the co-player's parameters and used higher-order derivatives. To overcome these shortcomings, Model-free Opponent Shaping (Lu et al., 2022b, M-FOS), proposed to frame shaping as a meta reinforcement learning problem. In M-FOS, the meta-agents are represented by deep reinforcement learning agents. The meta-agent learns to shape by observing full trainings of the co-players (a meta episode). At each meta-step, the meta-agent outputs the whole policy of a so-called inner agent (similar to HyperNetworks (Ha et al., 2017)) playing against co-players in an underlying game. Having to output whole policies limits M-FOS to games where the policy can be represented compactly, such as in matrix games. The original architecture is not suitable for games where policies are represented by deep neural networks. In fact, Lu et al. (2022b) avoid outputting whole neural

network parameterizations by simply outputting a conditioning variable for the inner agent, but only with modest success. Moreover, Lu et al. (2022b) train the meta-agents using deep reinforcement learning techniques in the cases where the policy is parameterized by deep neural networks. Such techniques are bound to fail as a meta-episode can contain millions of inner update steps long.

To address these issues, we propose Opponent Shaping via Slow Learning (EARL). In EARL, we combine the meta-agent and the inner agent into a single meta-agent parameterized by a recurrent neural network (RNN). The EARL agent meta-learns by retaining their hidden state throughout a whole meta-episode, thus capturing the learning dynamics of the co-player (similar to RL^2 (Duan et al., 2016). Combining the meta-agent and the inner agent into one recurrent meta-learner avoids outputting whole policy parameterizations. Furthermore, we demonstrate via evolutionary-based optimisers that EARL agents are scalable to longer meta-episodes and thus applicable to a much larger set of environments.

In our experiments, we show that EARL matches M-FOS agents' performance in the *infinitely* iterated prisoner's dilemma by converging to the ZD-extortion strategy. EARL also converges to ZD-extortion in the much more general *finitely* iterated prisoner's dilemma, the first method to reportedly do so. EARL also achieves much better shaping on Coin Game, a grid-based environment where policies are represented by deep neural networks.

To summarise our contributions

- Introduce EARL, a scalable shaping method suitable for higher dimensional games
- Demonstrate EARL converges to ZD-Extortion in sequential prisoners dilemma
- Demonstrate EARL exploits naive learners in Grid-World
- Introduce a new non-exploitable version of CoinGame suitable for shaping work
- Propose an experimental protocol to evaluate future shaping work

2 Related Work

Opponent Shaping Several methods recognise that their actions influence the learning of their opponents and take advantage of this to "shape" an opponent's policy. Most of these works assume white-box access to an opponent's parameters and reward to take higher-order derivatives through an opponent's update (Foerster et al., 2018a; Letcher et al., 2019a; Kim et al., 2021; Willi et al., 2022). Such methods are also myopic since anticipating many steps is intractable. Lu et al. (2022b) and Balaguer et al. (2022) solve the aforementioned issues by framing opponent shaping as a meta reinforcement learning problem. Yet they still are unable to scale to a large number of time steps and large policy parameterizations. To circumvent this, these approaches use the exact value function of the environment, so that inner episodes do not have to be rolled out, or output entire policies in a Hypernetwork-like manner. EARL inherits the benefits of the meta-learning formulation and improves on the shortcomings of Lu et al. (2022b) and Balaguer et al. (2022) by using a recurrent meta-learner.

Opponent Modeling Much work in MARL has focused on the idea of *opponent modelling*, in which an agent models some aspect of the policy of other agents in the environment. This includes explicitly modelling opponent policies (Mealing & Shapiro, 2017), modelling opponent intentions (**?**), classifying opponent strategies (Weber & Mateas, 2009; Synnaeve & Bessière, 2011), and modelling an opponent's nested beliefs (Wen et al., 2019). LILI (Xie et al., 2020) models an opponent's high-level latent strategy from local observations with a latent dynamics model rather than explicitly modeling the opponent's policy. Other work (**?**) has also considered learning effective policies in the presence of opponents with memory. However, these methods are not capable of actively *shaping* their opponents' learning dynamics. Thus they do not address the issue that we address in this paper.

Multi-Agent Meta-Learning M-FOS is a form of multi-agent meta-learning where a neural network parameterizes the meta-policy. Existing multi-agent meta-learning methods, such as Meta-Policy Gradient (Meta-PG) (**?**), Meta-MAPG (Kim et al., 2021), and Learning to Exploit (L2E) (Wu et al., 2021) instead parameterize the meta-policy using a method similar to that of Model-Agnostic Meta-Learning (MAML) (Finn et al., 2017a), in which they learn initial parameters and meta-learn

across their own gradient updates. While this type of meta-learning can adapt to any task at test time in single-agent settings (Xiong et al., 2021), in multi-agent settings, the calculated gradient may not correspond to a direction of improvement as the updates of other agents change the underlying dynamics. Rather than being restricted to a gradient update within the episode, M-FOS allows for arbitrary meta-policies that can carry out long-horizon opponent shaping.

3 BACKGROUND

Partially Observable Stochastic Game (POSG) A POSG is given by the tuple $= \langle n, \mathcal{A}, \mathcal{O}, S, \mathcal{T}, \mathcal{I}, \mathcal{R}, \rangle$, where $\mathcal{A}$, $\mathcal{O}$, and S denote the action, observation, and state space, respectively. These parameters can be distinct at every time step and also incorporated into the transition function $\mathcal{T} : S \times \mathbf{A} \rightarrow S$, where $\mathbf{A} \equiv \mathcal{A}^n$ is the joint action of all agents. Each agent draws individual observations according to the observation function $: S \times N \rightarrow \mathcal{O}$ and obtains a reward according to their reward function $: S \times \mathbf{A} \times N \rightarrow \mathbb{R}$ where $N = \{1, \ldots, n\}$. POSGs can represent general-sum games. The single player case, $\mathcal{I} = \{1\}$, of POSGs are Partially Observable Markov Decision Processes (POMDPs).

Model-Free Opponent Shaping (M-FOS) M-FOS frames opponent shaping as a meta reinforcement task. More specifically, the meta-task is formulated as a POMDP over an underlying general-sum game, represented by a POSG. In the POMDP, the meta-state spans the policies of every player in the underlying POSG. The meta-action space consists of the policy parameterization of the underlying inner agent playing the game for the meta-agent. At each meta-episode, conditioned on both agent's policies, M-FOS produces the parameters of the next inner-agents. The meta-reward is the return of the inner agent over one inner episode. Similar to most meta-reinforcement learning algorithms, M-FOS suffers from the credit assignment problem (Minsky, 1961) and thus limited to small inner timestep. To circumvent this, two approaches are used, depending on the underlying game. For matrix-games, meta-agents are trained with the exact gradients of the infinitely iterated version of the matrix-games (Foerster et al., 2018a). Here the meta-agent outputs the parameters of the inner-agent, and as such, is limited to simpler inner-agent policies. In grid-worlds, where getting an exact value function is intractable, M-FOS approximates the episode reward using REINFORCE (Williams, 1992), resulting in marginal performance improvements over baselines.

RL^2 In contrast to M-FOS HyperNetwork-like approach to meta-learning, meta-reinforcement learning solutions can be decomposed into two components: a 'fast' solution and a 'slow' solution. These components occur over two timescale. In RL^2 (Duan et al., 2016), the solution is encoded in the weights of the RNN Elman (1990), which are learned slowly through a general-purpose ('slow') RL algorithm. The RNN receives all information a typical RL algorithm would receive, including observations, actions, rewards, and termination flags; and it retains its state across episodes in a given Markov Decision Process (MDP). The activations of the RNN store the state of the "fast" RL algorithm on the current (previously unseen) MDP.

Evolution Strategies (ES) ES (Salimans et al., 2017, ES) is a derivative-free optimisation method. Let $F : \mathbb{R}^d \rightarrow \mathbb{R}$ be some function we want to optimise over. Instead of optimising $F(\mathbf{x})$ directly, we first blur the objective to

$$\mathbb{E}_{\epsilon \sim N(\mathbf{0}, I_d)}[F(\mathbf{x} + \sigma\epsilon)] \,,$$

where σ is a hyper-parameter dictating the amount of Gaussian noise we add. This is useful because

$$\nabla_{\mathbf{x}} \mathbb{E}_{\epsilon \sim N(\mathbf{0}, I_d)}[F(\mathbf{x} + \sigma\epsilon)] = \mathbb{E}_{\epsilon \sim N(\mathbf{0}, I_d)} \left[\frac{\epsilon}{\sigma} F(\mathbf{x} + \sigma\epsilon) \right] \,,$$

which allows us to optimise a non-differentiable function using gradient descent techniques in a highly scalable manner.

4 METHOD

Typically, existing shaping approaches are based on either MAML-like (Finn et al., 2017b), or HyperNetwork-like (Ha et al., 2017) approaches. MAML-like methods assume access to higher-order derivatives to directly shape the opponents' parameter update, requiring white-box access to

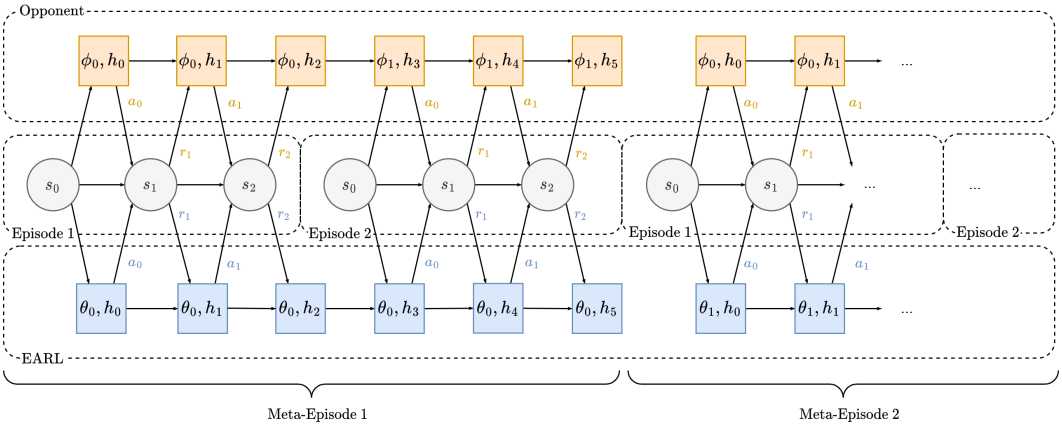

Figure 1: Example meta-learning interaction. EARL θ (blue) resets its hidden state h at the beginning of a meta-episode, updates its hidden state after each environment interaction, then updates its parameters at the end of a meta-episode. The opponent ϕ (orange) resets its parameters at the beginning of a meta-episode, then updates its parameters at the end of each episode. Optionally, if the opponent uses memory, its hidden state h is reset at the beginning of a meta-episode and updated after each environment interaction.

their differentiable learning algorithm (Foerster et al., 2018b; Letcher et al., 2019b; Willi et al., 2022). HyperNetwork-like approaches directly output the weights on the inner agent and thus can be optimised using model-free optimisation methods (Lu et al., 2022b; Balaguer et al., 2022). However, outputting the network weights requires a large action space, which suffers from the curse of dimensionality for more complex policies. Lastly, both approaches only shape the next learning steps instead of considering longer horizons.

Algorithm 1: General EARL

Initialise Shaper parameters θ.
while true **do**
  for $n \in \{1, \ldots, N\}$ **do**
    Sample shaper parameters θ_m from ES
    Initialise co-player parameters ϕ_0.
    for $t = 0$ **to** T **do**
      Reset Environment
      Gather trajectories τ_t given θ_n, ϕ_t
      Update ϕ_{t+1} according to respective learning algorithm
    end for
    Calculate fitness $\mathcal{F}_n$ for meta-episode
  end for
  Update ϕ with ES($\mathcal{F}$)
end while

Opponent shaping has previously been formulated as a meta-game. In a meta-game, the meta-state consists of the policies of all agents, a meta-step is an inner episode, the meta-reward is the inner return, and the meta-action is choosing the next inner policy, where "inner" refers to the underlying game. The key insight underlying Opponent Shaping via Slow Learning (EARL) is that we can represent both meta-policy and inner policy by the same recurrent neural network. Thus we learn an internal meta-state representation from the trajectories of actions observed so far. This policy can be optimised using model-free optimisation methods appropriate for sequential settings.

We formally construct a POSG $\mathcal{M} = \langle n, \bar{\mathcal{A}}, \bar{\mathcal{O}}, \bar{S}, \bar{\mathcal{T}}, \bar{\mathcal{I}}, \bar{\mathcal{R}}, \rangle$, over an underlying POSG $\mathcal{M}_k$. The meta-state space is given by $\bar{S} = S \times \{(h_1, ..., h_n)\}$, the Cartesian product of the underlying game state $\bar{S}$ and the internal representations of each player h_n. Similarly the meta-action space is given by $\bar{\mathcal{A}} = \mathcal{A} \times \{(h_1, ..., h_n)\}$, the Cartesian product of the underlying game state $\mathcal{A}$ and the internal representations of each player h_n. Unlike MAML-like or HyperNetwork-like approaches, the set of observations $\mathcal{O}$ and set of observation function $\mathcal{I}$ of the underlying game are used, which do not assume access to the opponent's internal state. The transition function $\mathcal{T}$, reward function $\mathcal{R}$ and the discount factor γ are constructed via the standard formalism.

Figure 1 illustrates the interaction between agents and the environment. We define the meta-game as a set of games $\{\mathcal{M}\}$ and a trial to be a sequence of k sampled games. At the start of a trial, n agents are initialised with internal states h_0^n and a new game $\mathcal{M}_0$ and observations o_0^n are drawn.

Upon receiving a state, agents generate actions as $a_t^n = \phi^n(o_t, h_t)$. At each time step, the internal state of each agent is updated such that $h_{t+1}^n = \psi^n(h_t^n, o_t^n)$. Once a game is terminated, a new game and initial state are drawn, yet the agent's internal state is preserved from the previous step. The objective is to maximize the expected total discounted reward accumulated during a single trial rather than a single episode.

We use evolutionary strategies (ES) to maximize the expected fitness across a population rather than the expected reward across a trajectory. At the start of each generation, a population of EARL agents of size M and N Naive Learners (NL) are initialized. During a generation, the population plays against copies of the Naive Learner in parallel for a series of k inner episodes. At the end of an inner episode, each copy of the Naive Learner performs a gradient update. At the end of the meta-episode, EARL performs a gradient update in the direction of the maximum expected fitness, and the next generation begins.

5 Experiments

5.1 Environments

Iterated Prisoner's Dilemma The prisoner's dilemma is a well-known and widely studied general-sum game, illustrating that two rational agents may not cooperate even if it appears to be in their best interest. Both players choose to either cooperate (C) or defect (D) and receive a payoff according to Table 1. The *iterated* prisoner's dilemma (IPD) is a game repeated by the same participants, where players can observe their opponent's past decisions.

Axelrod & Hamilton (1981) popularized the strategy Tit-For-Tat in their famous round-robin tournament. The strategy is to cooperate on the first turn and after that, choose the action last performed by its opponent. Tit-For-Tat can be considered a fixed mechanism to encourage mutual cooperation.

	C	D
C	(-1,-1)	(0, -3)
D	(-3, 0)	(-2, -2)

Table 1: Payoff matrix of IPD.

Beyond mutual cooperation, ZD-Extortion (Press & Dyson, 2012) elicits opponent cooperation while allowing the user to enforce a linear relationship that is more favourable than mutual cooperation. Notably, M-FOS learns ZD-Extortion in when playing against a Naive Learner (see 5.2).

Coin Game The Coin Game is a multi-agent, wrap-around grid-world environment that simulates social dilemmas (like the IPD) with high-dimensional dynamic states. Two players – blue and orange – move around a grid and pick up blue and orange coloured coins. When a player picks up its own colour coin, the player receives a reward of $+1$. When a player picks up a coin of the opponent's colour, the player receives a reward of $+1$ and the opponent receives a reward of -2. When both players pick up coins without regard to colour, the expected reward is 0. In contrast to the IPD, the Coin Game requires learning high-dimensional dynamic states across a longer time horizon, a task that current shaping methods struggle to learn.

5.2 Experimental Protocol

Competent Opponents We define an agent as *competent* if its strategy reaches the Nash equilibrium. In IPD, this is choosing to always defect; in the Coin Game, this is picking up as many coins as quickly as possible without regard to colour.

In the Coin Game, we discover that agents cannot learn (via reinforcement learning) when trained against a competent opponent. To address this, we show that adjusting the observation that an agent receives to an *egocentric* viewpoint (i.e. an agent always observes that it

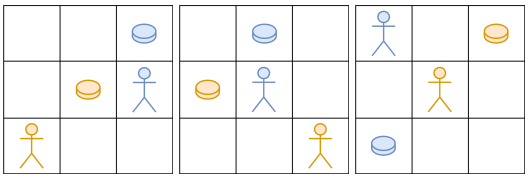

Figure 2: Illustration of (a) a state observed by both agents in the non-egocentric Coin Game and (b)-(c) the same state observed by the blue and orange agent, respectively, in the egocentric Coin Game.

is in the centre of the grid) leads to competency against a competent opponent. Throughout the rest of the paper, we refer to the original version of Coin Game as *non-egocentric* Coin Game and the modified observation version as *egocentric* Coin Game.

Experimental Protocol We propose an experimental protocol for producing the correct conditions for shaping. First, we verify that an agent can learn against three competent opponents. The first agent is *Good* Greedy, picking up coins indiscriminately but prioritizing its own colour coin if it is equidistant from two coins; the second agent is *Evil* Greedy, picking up coins indiscriminately but prioritizing its opponent's coin if it is equidistant from two coins; the third agent is the current agent pre-trained to competency via self-play. After verifying that an agent learns competency against these three opponents, we set the parameters of the meta-episode to reflect the time and scale required for the agent to become competent against a competent opponent.

5.3 Baseline Comparisons

Tabular A Tabular agent uses a look-up table with observations as input and actions as outputs. Importantly, this agent is *not* parameterized by a neural network. Previous work shows that shaping through Adversarial Cheap Talk (ACT) (Lu et al., 2022a) only works if the shaped agent is parameterized by a neural network. The Tabular agent baseline distinguishes if EARL shapes through ACT or through another mechanism.

Naive Learner A Naive Learner (NL) agent assumes other agents are part of the environment. It updates at the end of each inner episode with learning rate α:

$$\phi^i_{t+1} = \phi^i_t + \alpha \nabla_{\phi^i_t} \mathcal{R}^i(\phi^i_t, \phi^{-i}_t) \tag{1}$$

In the IPD, NL is parameterized as a feed-forward network trained using PPO (Schulman et al., 2017). In the Coin Game, the NL is parameterized by a recurrent neural network trained using PPO.

Model-Free Opponent Shaping (M-FOS) An M-FOS agent uses model-free optimization techniques to train a meta-policy whose meta-actions are choosing the next inner policy of the underlying game. In the IPD and Coin Game, the M-FOS inner policy is parameterized as a recurrent neural network that inputs observations and a conditioning vector from the meta-policy. M-FOS meta-policy is parameterized by a recurrent neural network that inputs the previous episode's trajectory and outputs a conditioning vector used by the M-FOS inner policy.

5.4 Training Details

Unlike M-FOS, an EARL agent's policy acts as both the mechanism designer and the mechanism. Since there is no distinction between the meta-policy and the inner policy, EARL can be played in different POSGs without complex architecture changes. In the IPD and Coin Game, EARL is parameterized as a recurrent neural network and is trained using OpenES (Salimans et al., 2017).

6 Results

6.1 Iterated Prisoner's Dilemma

	EARL	M-FOS	PPO
EARL	(N/A, N/A)	(N/A, N/A)	**(-0.23, -2.71)**
M-FOS	(N/A, N/A)	(N/A, N/A)	(N/A, N/A)
PPO	(-2.71, -0.23)	(N/A, N/A)	(-2.00, -2.00)

Table 2: Head-to-head results in the *finite* IPD showing each agent's average reward per timestep.

EARL v. PPO In the IPD, EARL achieves a score of of -0.23, achieving ZD-Extortion by shaping PPO to cooperate almost unconditionally. M-FOS also reaches ZD-Extortion with a score of -0.51; however, it does so by calculating $\mathcal{R}$ and its gradient exactly. In general, this technique

can only be applied to matrix games, while sample-based approaches like policy gradients or evolutionary strategies can be applied to any general-sum game. Notably, EARL outperforms M-FOS without requiring exact values or gradients.

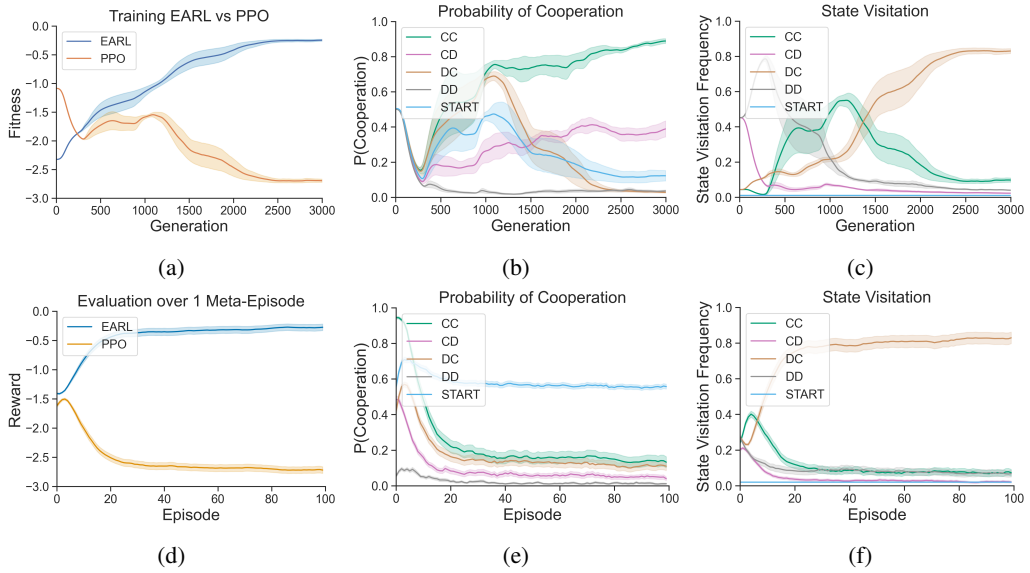

Figure 3: Training results in the finite IPD over 5 seeds. (a) Fitness (b) EARL's probability of cooperation conditioned on state and (c) state visitation. Notice that EARL achieves a score higher than the payoff for mutual cooperation of -1 and that the state visitations for DC reach 80% after 2000 generations of training, indicating that EARL has learned to extort its opponent. Evaluation results over a single meta-episode composed of 100 inner episodes over 20 seeds. (d) Reward (e) EARL's probability of cooperation conditioned on state and (f) state visitation. Notice that EARL starts by playing a fully cooperative strategy, then immediately starts to anneal its probability of cooperation until episode 20, where it is then playing with a fully extortionist strategy.

6.2 Coin Game

Sanity We verify that PPO RNN can learn against competent opponents in the Coin Game in Figure 4. PPO RNN learns to pick up the same number of coins as a pre-trained PPO RNN, Good Greedy, and Evil Greedy. These hyperparameters are the same ones used during meta-learning.

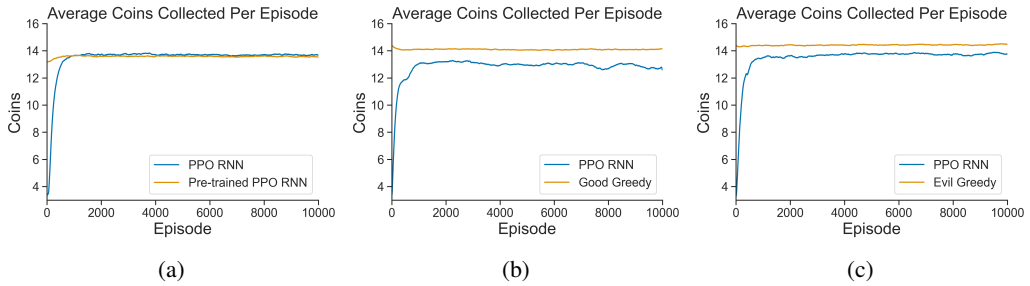

Figure 4: Results of the experimental protocol verifying that PPO RNN learns to play the egocentric Coin Game against (a) pre-trained PPO RNN (b) Good Greedy (c) Evil Greedy. Notice that the agent learns to pick up roughly the same number of coins per episode as its competent opponents.

EARL v. Tabular EARL learns to pick up 13 coins per episode and vastly outperforms the Tabular agent (6.67, -6.51). As EARL becomes more competent, the Tabular agent picks up slightly fewer coins per episode, starting at 9 coins per episode and dropping to 8 coins per episode by the

	EARL	M-FOS	PPO RNN	PPO	Tabular
EARL	(N/A, N/A)	(N/A, N/A)	**(4.04, 1.12)**	**(4.45, 2.69)**	**(6.67, -6.51)**
M-FOS	(N/A, N/A)	(N/A, N/A)	(2.01, -1.82)	(0.94, 0.04)	(2.51, -3.03)
PPO RNN	(1.12, 4.04)	(-1.82, 2.01)	(0.38, 0.30)	(-0.21, 0.12)	(7.08, -7.36)
PPO	(2.69, 4.45)	(0.04, 0.94)	(0.12, -0.21)	(0.21, 0.20)	(1.48, -1.58)
Tabular	(-6.51, 6.67)	(-3.03, 2.51)	(-7.36, 7.08)	(-1.58, 1.48)	(-0.31, -0.30)

Table 3: Head-to-head results in the *egocentric* Coin Game showing each agent's reward per episode.

end of training. While staying competent at Coin Game, the Tabular agent is shaped into preferentially picking up its own colour coin 62% of the time, indicating that EARL does *not* shape through ACT. Against the tabular agent, EARL outperforms M-FOS.

EARL v. PPO Although EARL averages 10 coins per episode versus PPO's 12 coins per episode, EARL outperforms PPO $(4.45, 2.69)$ because EARL has shaped PPO into preferentially picking up its own colour coins 77% of the time, leading to fewer negative rewards for EARL. EARL outperforms M-FOS in the matchup against PPO in both absolute reward and in the sum of reward across both players. The latter result suggests that EARL is more pro-social in its behavior, shaping its opponent to be more cooperative without negatively impacting the sum of rewards of both players.

EARL v. PPO RNN Against PPO RNN, EARL performs similarly to its matchup against PPO. It shapes PPO RNN to to preferentially pick up its own coins 70% of the time, resulting in better performance over PPO RNN $(4.04, 1.12)$. Over the course of training, EARL reduces the competency of PPO RNN minimally from 13 coins per episode to 12 coins per episode as shown in (c) Figure 5. Like in the matchup against PPO, EARL outperforms M-FOS in the matchup against PPO RNN in both absolute score and in the sum of rewards across both players, suggesting that EARL learns to shape the behavior of its opponent that is both beneficial to itself and to both players.

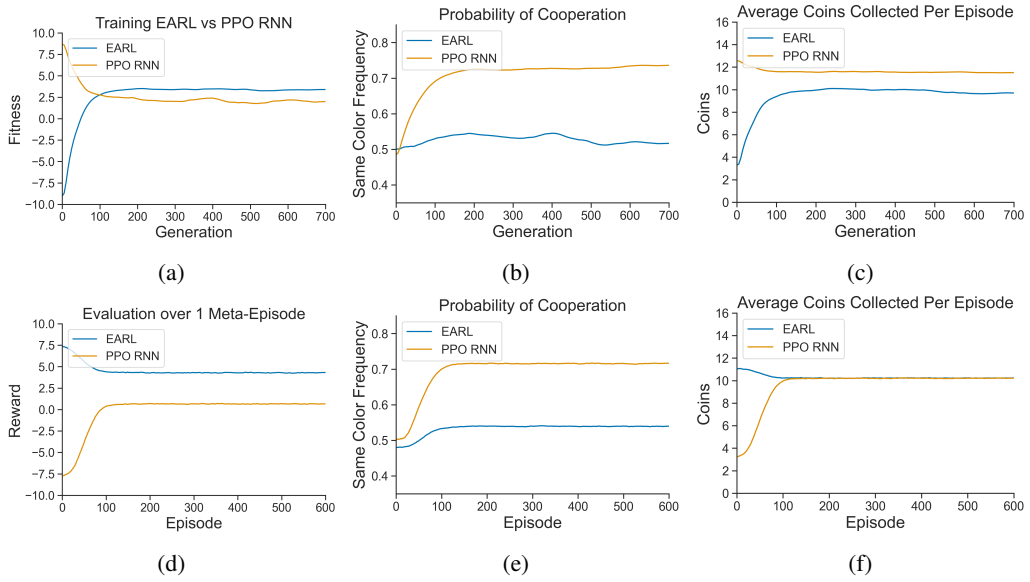

Figure 5: Training results of EARL vs. PPO RNN in the egocentric Coin Game. (a) Fitness, (b) both agent's frequency of picking up its own colour coin, (c) the number of coins picked up per episode. Notice that EARL learns to be competent at the game while shaping the opponent into preferentially picking up its own colour coins. Evaluation run of a single meta-episode of 600 inner episodes. (d) Reward, (e) both agent's frequency of picking up its own color coin and (f) the number of coins picked up per episode. Note how the opponent's policies are shaped into preferentially picking up its own coin 70% of the time, resulting in a higher score for EARL, despite both players picking up the same number of coins.

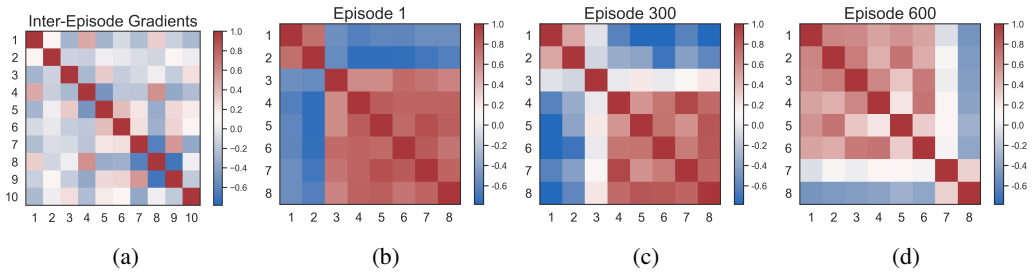

Figure 6: (a) Inter-episode cosine similarity of the gradients over a single meta-episode composed of 600 inner episodes across 10 bins. The (b) first, (c) middle, and (d) last intra-episode cosine similarity of the gradients over a single inner episode composed of 16 environment timesteps over 8 bins.

7 Conclusion

To summarise our contributions

- Introduce EARL, a scalable shaping method suitable for higher dimensional games
- Demonstrate EARL converges to ZD-Extortion in sequential prisoners dilemma
- Demonstrate EARL exploits naive learners in Grid-World
- Introduce a new non-exploitable version of CoinGame suitable for shaping work
- Propose an experimental protocol to evaluate future shaping work

In the future, we would like to expand the work in three ways: increase the number of opponents that we play against, increase the complexity of the environments (Leibo et al., 2021), and increase the difficulty of the opponents.

8 Acknowledgements

References

Robert Axelrod and William D Hamilton. The evolution of cooperation. *science*, 211(4489):1390–1396, 1981. 1, 5

Jan Balaguer, Raphael Koster, Christopher Summerfield, and Andrea Tacchetti. The good shepherd: An oracle agent for mechanism design. *CoRR*, abs/2202.10135, 2022. URL `https://arxiv.org/abs/2202.10135`. 1, 2, 4

Yan Duan, John Schulman, Xi Chen, Peter L. Bartlett, Ilya Sutskever, and Pieter Abbeel. Rl^2: Fast reinforcement learning via slow reinforcement learning. arXiv preprint arXiv:1611.02779, 2016. 2, 3

Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990. 3

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1126–1135, 2017a. 2

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pp. 1126–1135. PMLR, 2017b. 3

Jakob Foerster, Richard Y Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch. Learning with opponent-learning awareness. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 122–130, 2018a. 1, 2, 3

Jakob N. Foerster, Gregory Farquhar, Maruan Al-Shedivat, Tim Rocktäschel, Eric P. Xing, and Shimon Whiteson. Dice: The infinitely differentiable monte carlo estimator. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1524–1533, 2018b. 4

David Ha, Andrew M. Dai, and Quoc V. Le. Hypernetworks. In *5th International Conference on Learning Representations*, 2017. 1, 3

Marc Harper, Vincent Knight, Martin Jones, Georgios Koutsovoulos, Nikoleta E. Glynatsi, and Owen Campbell. Reinforcement learning produces dominant strategies for the iterated prisoner's dilemma. *PLOS ONE*, 12(12):e0188046, 2017. 1

Max Jaderberg, Wojciech M. Czarnecki, Iain Dunning, Luke Marris, Guy Lever, Antonio Garcia Castañeda, Charles Beattie, Neil C. Rabinowitz, Ari S. Morcos, Avraham Ruderman, Nicolas Sonnerat, Tim Green, Louise Deason, Joel Z. Leibo, David Silver, Demis Hassabis, Koray Kavukcuoglu, and Thore Graepel. Human-level performance in 3d multiplayer games with population-based reinforcement learning. *Science*, 364(6443):859–865, 2019. doi: 10.1126/science.aau6249. 1

Dong-Ki Kim, Miao Liu, Matthew Riemer, Chuangchuang Sun, Marwa Abdulhai, Golnaz Habibi, Sebastian Lopez-Cot, Gerald Tesauro, and Jonathan P. How. A policy gradient algorithm for learning to learn in multiagent reinforcement learning. In *International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 5541–5550, 2021. 2

Joel Z. Leibo, Edgar A. Duéñez-Guzmán, Alexander Vezhnevets, John P. Agapiou, Peter Sunehag, Raphael Koster, Jayd Matyas, Charlie Beattie, Igor Mordatch, and Thore Graepel. Scalable evaluation of multi-agent reinforcement learning with melting pot. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 6187–6199. PMLR, 2021. 9

Alistair Letcher, David Balduzzi, Sébastien Racanière, James Martens, Jakob N. Foerster, Karl Tuyls, and Thore Graepel. Differentiable game mechanics. *J. Mach. Learn. Res.*, 20:84:1–84:40, 2019a. 2

Alistair Letcher, Jakob N. Foerster, David Balduzzi, Tim Rocktäschel, and Shimon Whiteson. Stable opponent shaping in differentiable games. In *7th International Conference on Learning Representations*, 2019b. 1, 4

Chris Lu, Timon Willi, Alistair Letcher, and Jakob Nicolaus Foerster. Adversarial cheap talk. In *Decision Awareness in Reinforcement Learning Workshop at ICML 2022*, 2022a. 6

Christopher Lu, Timon Willi, Christian A Schroeder De Witt, and Jakob Foerster. Model-free opponent shaping. In *International Conference on Machine Learning*, pp. 14398–14411. PMLR, 2022b. 1, 2, 4

Richard Mealing and Jonathan L. Shapiro. Opponent modeling by expectation-maximization and sequence prediction in simplified poker. *IEEE Trans. Comput. Intell. AI Games*, 9(1):11–24, 2017. 2

Marvin Minsky. Steps toward artificial intelligence. *Proceedings of the IRE*, 49(1):8–30, 1961. 3

William H. Press and Freeman J. Dyson. Iterated Prisoner's Dilemma contains strategies that dominate any evolutionary opponent. *Proceedings of the National Academy of Sciences*, 109(26): 10409–10413, 2012. ISSN 0027-8424. 5

Tim Salimans, Jonathan Ho, Xi Chen, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. arXiv preprint arXiv:1703.03864, 2017. 3, 6

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347, 2017. 6

DJ Strouse, Kevin R. McKee, Matt M. Botvinick, Edward Hughes, and Richard Everett. Collaborating with humans without human data. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 14502–14515, 2021. 1

Gabriel Synnaeve and Pierre Bessière. A bayesian model for opening prediction in RTS games with application to starcraft. In *IEEE Conference on Computational Intelligence and Games*, pp. 281–288, 2011. 2

Ben George Weber and Michael Mateas. A data mining approach to strategy prediction. In *Proceedings of the 2009 IEEE Symposium on Computational Intelligence and Games*, pp. 140–147, 2009. 2

Ying Wen, Yaodong Yang, Rui Luo, Jun Wang, and Wei Pan. Probabilistic recursive reasoning for multi-agent reinforcement learning. In *7th International Conference on Learning Representations*, 2019. 2

Timon Willi, Alistair Letcher, Johannes Treutlein, and Jakob N. Foerster. COLA: consistent learning with opponent-learning awareness. In *International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 23804–23831, 2022. 1, 2, 4

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8:229–256, 1992. 3

Zhe Wu, Kai Li, Enmin Zhao, Hang Xu, Meng Zhang, Haobo Fu, Bo An, and Junliang Xing. L2E: learning to exploit your opponent. arXiv preprint arXiv:2102.09381, 2021. 2

Annie Xie, Dylan P. Losey, Ryan Tolsma, Chelsea Finn, and Dorsa Sadigh. Learning latent representations to influence multi-agent interaction. In *4th Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pp. 575–588, 2020. 2

Zheng Xiong, Luisa M. Zintgraf, Jacob Beck, Risto Vuorio, and Shimon Whiteson. On the practical consistency of meta-reinforcement learning algorithms. *CoRR*, abs/2112.00478, 2021. URL `https://arxiv.org/abs/2112.00478`. 3

A Appendix

You may include other additional sections here.

A.1 EARL

A.2 Naive Learner

Hyperparameter	Value
Number of Actor Hidden Layers	1
Size of Actor Hidden Layers	[25]
Number of Critic Hidden Layers	1
Size of Critic Hidden Layers	[25]
Length of Meta-Episode T	100
Batch Size B	100
Number of Epochs	2
Outer Discount Factor γ	0.96
PPO Clipping ϵ	0.2
Entropy Coefficient	0.01
Size of Hidden Layer	25
Initial Scale of Isotropic Gaussian Noise	0.04
Multiplicative Decay Factor	0.999
Value Coefficient	0.5
Sigma Limit	0.01
Minimum mean initialization	0.0

Table 4: Hyperparameters for EARL in Iterated Prisoner's Dilemma