The Benefits of Power Regularization in Cooperative Reinforcement Learning

Michelle Li
MIT
Boston, USA
limichelle.lm@gmail.com

Michael Dennis
University of California, Berkeley
Berkeley, USA
michael_dennis@cs.berkeley.edu

ABSTRACT

Cooperative Multi-Agent Reinforcement Learning (MARL) algorithms, trained only to optimize task reward, can lead to a concentration of power where the failure or adversarial intent of a single agent could decimate the reward of every agent in the system. In the context of teams of people, it is often useful to explicitly consider how power is distributed to ensure no person becomes a single point of failure. Here, we argue that explicitly regularizing the concentration of power in cooperative RL systems can result in systems which are more robust to single agent failure, adversarial attacks, and incentive changes of co-players. To this end, we define a practical pairwise measure of power that captures the ability of any co-player to influence the ego agent's reward, and then propose a power-regularized objective which balances task reward and power concentration. Given this new objective, we show that there always exists an equilibrium where every agent is playing a power-regularized best-response balancing power and task reward. Moreover, we present two algorithms for training agents towards this power-regularized objective: Sample Based Power Regularization (SBPR), which injects adversarial data during training; and Power Regularization via Intrinsic Motivation (PRIM), which adds an intrinsic motivation to regulate power to the training objective. Our experiments demonstrate that both algorithms successfully balance task reward and power, leading to lower power behavior than the baseline of task-only reward and avoid catastrophic events in case an agent in the system goes off-policy.

KEYWORDS

Multi-Agent Reinforcement Learning; Cooperative Multi-Agent Reinforcement Learning; Game Theory; Intrinsic Motivation; Fault Tolerance; Adversarial Robustness; Distribution of Power

ACM Reference Format:
Michelle Li and Michael Dennis. 2023. The Benefits of Power Regularization in Cooperative Reinforcement Learning. In *Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), London, United Kingdom, May 29 – June 2, 2023*, IFAAMAS, 9 pages.

1 INTRODUCTION

When considering how to optimally structure a team, institution, or society, a key question is how responsibility, power, and blame ought to be distributed, and as such it is a broadly studied concept in the social sciences [21, 22]. We often want to avoid too much power lying in the hands of a few actors, preferring power to be distributed rather than concentrated. One example of a distributed system of power is the US government: in principle, having three branches with checks and balances helps prevent any single branch from having too much power.

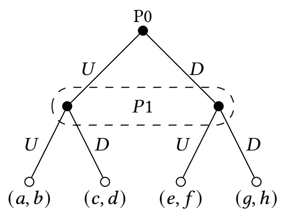
The same basic idea applies in Multi-Agent Reinforcement Learning (MARL) systems: regardless of whether the setting is fully cooperative, fully competitive, or general sum, it is often advantageous for agents to limit the amount of power other agents have.

Power resists formalization despite being a prevalent and intuitive concept. In this paper, we make no normative claims about how power ought to be defined or regulated, just that power is a conceptually useful tool for cooperative multi agent systems. To focus on empirical progress, we limit our attention to power as *influence on reward*. But regardless of how it is formalized, we argue that avoiding concentration of power can simultaneously mitigate three problems that are generally thought of separately: system failure, adversarial attacks, and incentive changes. It does so by mitigating the effects of off policy behavior, regardless of the cause.

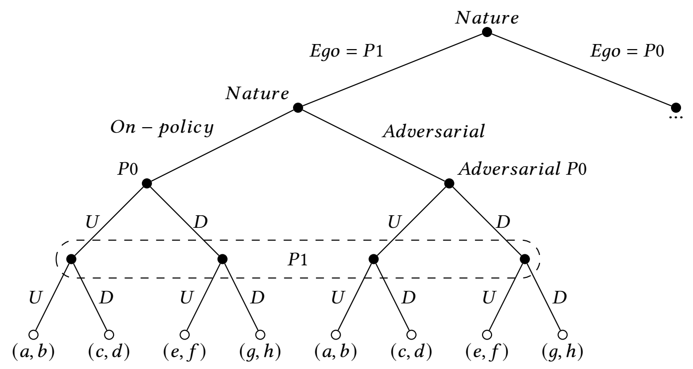
Consider the following toy example: a group of agents need to work cooperatively to maximize production. Agents produce output by starting from raw material and applying a series of m actions. They can either work individually or form an assembly line, which is more efficient because of specialization and batch productivity but requires that every agent is a single point of failure. Changes in any agent's behavior would bring the whole system to a halt. Depending on how much we care about task reward versus robustness, we might prefer one behavior or the other.

Our contributions in this work are as follows: 1) We propose a practical measure of power amenable to optimization – how much another agent can decrease our return by changing their action for one timestep. 2) We propose a framework for balancing maximizing task reward and minimizing power by regularizing the task objective for power, and then show an equilibrium always exists with this modified objective. 3) We present two algorithms for achieving power regularization: one, Sample Based Power Regularization (SBPR), which injects adversarial data during training by adversarially perturbing one agent's actions with some probability at any timestep; and two, Power Regularization via Intrinsic Motivation (PRIM), which adds an intrinsic reward to regularize power at each timestep. SBPR is simpler but PRIM is better able to achieve the right reward-power tradeoffs for very small values of λ. Our experiments in an Overcooked-inspired environment [3] demonstrate that both algorithms can achieve various power-reward tradeoffs and can reduce power compared to the task reward-only baseline.

Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), A. Ricci, W. Yeoh, N. Agmon, B. An (eds.), May 29 – June 2, 2023, London, United Kingdom. © 2023 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.



(a) Simple general-sum game



(b) p-Adversarial Game of (a)

Figure 1: Extensive Form diagrams of a simple 1 timestep game and its corresponding p-Adversarial game at s where Nature chooses an ego agent and whether to use the on-policy or adversarial co-player toward the ego agent. The dashed lines encircle nodes in the same information set for Player 1 because agents act simultaneously. We omit a final layer of Nature nodes that model the probabilistic transition function.

2 RELATED WORK

Power has been broadly studied in the social sciences [21, 22] and for multi-agent systems, but has not been studied in the context of deep multi-agent RL to the best of our knowledge. Instead, prior work has focused on graph-theoretic analysis [10] or symbolic formulations [5]. There have also been productive formulations of the related concepts of responsibility and blame [1, 4, 7, 8, 11], which have strong connections to power.

In AI, power has been formalized in a single agent context, with recent work towards defining power and regularizing an agent's own behavior with respect to power [27–29]. While it is a promising direction to extend these formal measures to MARL, we focus on making empirical progress on regularizing power in this work.

Though the literature on power in deep MARL is sparse, the literature on the problems that power regulation can help mitigate is more robust. For instance, there is a large body of work on designing MARL systems that cooperate robustly in sequential social dilemmas [6, 14, 15, 17], in which balancing power amongst the agents is critical. There is also a significant body of work showing the existence of adversarial attacks for neural networks [25] and single- and multi-agent RL [9, 13, 16, 18]. In such cases, power regularization can help build fault tolerance.

Finally, the algorithms we present for power regularization may be well suited for situations where one desires agents that work well with unknown teammates, if combined with approaches from ad-hoc teamwork [2, 24] and zero-shot coordination [12, 26]. This is especially true in situations where agents must infer who to trust (i.e. who to entrust power to) [20, 23].

3 BACKGROUND

We model our setting as a Markov game [19] defined by the tuple (N, S, A, T, R, γ) where N is the number of players, S is the set of environment states, $A = \times_{i \in \{0,...,N\}} A_i$ is the joint action space, $T : S \times A \rightarrow S$ is the transition function, $R = \times_{i \in \{0,...,N\}} R_i$ is the reward function for each player, and $\gamma \in (0, 1]$ is the discount factor. At every timestep, each agent i chooses an action $a_i \in A_i$, and the joint action $a = (a_1, a_2, ..., a_n) \in A$ is used to transition the environment according to $T(s'|a, s)$. Each agent i receives their own reward $r_i = R_i(s, a)$. While our theoretical results hold for general-sum Markov games, we only empirically evaluate on fully cooperative environments, thus assuming that all agents have the same reward: $R_i = R_j \; \forall i, j$. We operate in the fully-observed setting: each agent can directly observe the true state s. Each agent i aims to independently maximize their time-discounted, expected reward $U_i = \mathbb{E}[\sum_{t=0}^{T} \gamma^t r_t]$ where r_t is the reward at time t. Throughout, we will assume finite, discrete actions and finite time.

In any game, it is useful for a player to consider their *best responses*: optimal policies given fixed policies for the co-players:

$$\pi_i \in \underset{\pi'_i \in \Pi_i}{\arg\max}\{U_i(\pi'_i; \pi_{-i})\} = BR(\pi_{-i}).$$

where π_{-i} refers to the policies for all players other than i. Furthermore, we say that a policy π_i is a *local best response* at state s if it chooses an optimal distribution of actions at s given the rest of its policy π_i and co-player policies π_{-i}. That is,

$$\pi_i \in \underset{\pi'_i \in \overline{\Pi}_i(\pi_i; s)}{\arg\max} \{U_i(\pi'_i; \pi_{-i})\} = BR^{local}(\pi_{-i}).$$

where $\overline{\Pi}_i(\pi_i; s) = \{\pi'_i \in \Pi_i | \forall s' \neq s, \pi'_i(s') = \pi(s')\}$

A set of policies π for each player where each policy π_i is a best response to the other strategies is called a Nash equilibrium that is, $\forall i, \pi_i \in BR(\pi_{-i})$). This is a "stable point" where no agent has an incentive to unilaterally change their policy. Furthermore, we say that a set of policies π form a *local Nash equilibrium* at state s if all policies are a local best-response to the other policies at s.

4 FORMALISM

To design systems that regularize for power, it is important to be clear about our objective and how we define power.

Table 1: An example game where player 1 has no power over player 2 because all of 1's actions are equally bad for 2.

	X	Y	Z
X	$(3, -10)$	$(3, -9)$	$(3, -8)$
Y	$(2, -10)$	$(2, -9)$	$(2, -8)$
Z	$(1, -10)$	$(1, -9)$	$(1, -8)$

4.1 Measuring Power

Our main goal is to make empirical progress on building power-regularizing MARL systems, so we will not aim to find the most proper or most general definition of power. We define a measure for power of co-player j over ego agent i, which we call 1-step adversarial power, as the difference j could make on i's reward if j had instead acted adversarially toward i for one timestep.

Definition 4.1 (1-step adversarial power). Let U_i^{task} denote player i's task utility. Given policies π, the 1-step adversarial power agent j has on agent i when starting from state s is:

$$\texttt{power}(i, j|s, \pi) = r + \mathbb{E}[U_i^{\text{task}}(s', \pi)] - \min_{a_j \in A_j}(r_{a_j} + \mathbb{E}[U_i^{\text{task}}(s'_{a_j}, \pi)])$$

where $s' = T(s, \pi)$, $s'_{a_j} = T(s, a_j; \pi_{-j})$, and r and r_{a_j} are i's rewards obtained on-policy and with j's deviation to a_j, respectively.

The $\min_{a_j \in A_j}$ is taken over the set of deterministic actions. Note that it is not necessary to consider stochastic policies as the most powerful stochastic π_j could simply place probability 1 on any of the deterministic actions that achieve the lowest utility for i.

Counterintuitively, it is possible that all of j's immediate actions exert some causal effect on i's utility without j having any power over i. This can happen if all of j's actions reduce i's reward by 10, for example. Since we are defining power in relative terms, if all actions have the same effect on i, we say j has no power over i as any causal effects of j's actions on i are inevitable. See Table 1 for an example worked out explicitly. Such nuance is reminiscent of the difficulties in defining blame [4]. Exploring such connections in-depth could be a path towards better metrics for measuring power.

4.2 Regularizing for Power

Traditionally, cooperative MARL algorithms aim to optimize the discounted sum of task rewards, which we call *task utility*, without explicit consideration for the amount of power held by other agents. We argue that we can make systems more robust by optimizing an explicit trade-off between maximizing task reward and minimizing power. This framework has the advantage of addressing system failure, adversarial attacks, and incentive changes all at once by mitigating the negative effects of off-policy behavior.

We focus on linear trade-offs, that is objectives of the form

$$U_i(\pi|s) = U_i^{\text{task}}(\pi|s) + \lambda U_i^{\text{power}}(\pi|s) \tag{1}$$

where $U_i^{\text{task}}(\pi|s)$ is the task utility for player i starting in state s and $U_i^{\text{power}}(\pi|s) = \sum_{t=0}^{T} R_i^{power}(s_t, \pi)$ is the sum of power rewards R_i^{power} at states starting from s reached by unrolling π. In the 2-agent setting, $R_i^{power}(s, \pi) = -\texttt{power}(i, j|s, \pi)$, but with more agents, R_{power} must aggregate information about the powers of all j on i. In our experiments with more than 2 agents we let $R_i^{power}(s, \pi) = -\frac{1}{N-1}\sum_j \texttt{power}(i, j|s, \pi)$ (the mean function). We leave the problem of determining the most appropriate choice of aggregation function to future work.

Consider the 2-player general-sum matrix game defined in Table 2 which we call the Attack-Defense game. This is a single timestep game so $U_i^{\text{power}}(\pi|s) = -\texttt{power}(i, j|s, \pi)$. If either player plays X, the other player could play Z reducing the utility. playing Y to guarantees 2 utility, paying a small price to reduce power.

If optimizing purely for task reward, both agents play X to achieve the utility-maximizing Nash equilibrium. However, playing X incurs 3 power while playing Y incurs 0 power. Thus, by Eq 1 we have $U_{PR}(X) = 3 - 3\lambda$ and $U_{PR}(Y) = 2$, so for $\lambda > \frac{1}{3}$ we prefer Y. See Table 3 for a larger variant of the game with a nontrivial Pareto frontier. Figure 2a shows the value of each action as a function of λ assuming the coplayer is on-policy (i.e. doesn't play F).

It is important for the regularized objective to apply at every state, even those unreachable on policy. A naive approach to penalizing power would be to only penalize the 1-step adversarial power over the agent in the initial state, that is, only aiming to maximize $U_i(\pi|s_0)$ where s_0 is the initial state. However, such a measure has a fundamental flaw, in that once an agent deviates from the usual strategy, there is no longer any incentive to regulate power and thus our agent would *gain trust in potentially adversarial coplayers.*

For instance, suppose the optimal power regularized policy were to work independently instead of forming an assembly line. Once an agent deviates, the system could be in some state s not reachable on-policy, only reachable via an adversarial deviation a. The only way behavior in state s influences the utility at the initial state $U_i(\pi|s_0)$ is through the adversarial action term of the power regularization $U_i^{\text{power}}(\pi|s_0)$. Since this term increases when task reward increases, after any deviation the policy will no longer regulate power. Thus once one agent fails, all agents would revert to forming brittle and power-concentrating assembly lines. This is the opposite of the desired behavior: we would take a failure of an agent as an indication that they should be entrusted with more power because our model does not allow them to deviate again.

Luckily, the state conditioned regularization we propose is a simple fix. Rather than regularizing for power just at the first state, we regularize power via optimizing Eq 1 at all s. Thus even after an agent fails others will still continue to regularize for power.

5 EXISTENCE OF EQUILIBRIA

In the standard formulation of Markov Games, the existence of an equilibrium solution is guaranteed by Nash's Theorem, which shows that every finite game has a mixed Nash equilibrium. However, once we regularize for power, Nash's theorem no longer applies because the payoff becomes a function of the strategy.

Given our power-regularized objective, we can define notions of best response and equilibrium similar to the standard formulations.

Definition 5.1. We say that a policy π_i is a λ-power regularized best response to the policies π_{-i}, notated as $\pi_i \in PRBR_\lambda(\pi_{-i})$, if it achieves the optimal trade off between task reward and power minimization in every state. That is, for all s we have:

$$\pi_i \in \arg\max_{\pi'_i \in \Pi_i}\{U_i(\pi'_i; \pi_{-i}|s)\} = PRBR_\lambda(\pi_{-i}|s).$$

Table 2: The Attack-Defense Game: an opponent can take away your utility if you play X, but you can pay a small cost to defend against that by playing Y.

	X	Y	Z
X	$(3,3)$	$(3,2)$	$(0,0)$
Y	$(2,3)$	$(2,2)$	$(2,0)$
Z	$(0,0)$	$(0,2)$	$(0,0)$

Next, we define Power Regularizing Equilibrium (PRE) to be a fixed point of the power regularized best response function and then prove they are guaranteed to exist in any game.

Definition 5.2 (λ-Power Regularizing Equilibrium). A λ-Power Regularizing Equilibrium (PRE) is a policy tuple π such that all policies are power regularized best responses to the others. That is, in every state s, for all i, we have $\pi_i \in PRBR_\lambda(\pi_{-i}|s)$.

Theorem 5.3. *Let G be a finite, discrete Markov Game, then a λ-power regularizing equilibrium exists for any λ.*

Intuitively, we prove this by constructing another game, which we call the p-adversarial game of G, to which Nash's theorem can be applied, and show that Nash equilibria in this modified game correspond to power regularizing equilibria in the original game. The basic idea of this game is to add adversarial players that perturb co-players' actions adversarially toward the ego agent with probability p. We define the p-adversarial game formally below and depict the extensive form of a 1-timestep game in Figure 1b.

Definition 5.4 (p-Adversarial Game of G at s). The p-adversarial game of G at state s adds adversarial agents $\pi_j^{A^*i}$ for each player i and co-player j. These agents are randomly given control of i's co-players to minimize i's return. The game starts at s. Nature randomly decides with probability p to let the adversary will take control in this episode. Nature uniformly randomly selects an ego-agent i which will be the only agent to be rewarded in the game, and uniformly randomly selects a time step on which the adversary will take control, if the adversary gets control this episode. At that time step Nature will, choose a co-player j to be replaced by $\pi_j^{A^*i}$ for one step. Rewards for i are calculated as normal.

The following theorem establishes a correspondence between the best response in the p-adversarial game of G and the power regularized best response in the original game. We will use this correspondence to prove Theorem 5.3.

Theorem 5.5. *Consider an agent i in a Markov game G with time horizon T, and an arbitrary state s. The utility of policies in the p-adversarial game at state s, for $p = \lambda$ is equivalent to the corresponding policies in the original game. That is, for all π in the original Markov game, we have: $U_i^{p\text{-}Adv}(\pi_i; \pi_{-i}|s) = U_i(\pi_i; \pi_{-i}|s)$.*

The proof idea is to, for a p-adversarial game on a fixed state s, inductively argue for the equivalence starting from the last step of the game. At each step, the set of $\pi_j^{A^*i}$ for all co-players j can be seen as a way to compute i's power regularization term. Throughout the proof we use state-based rewards to simplify the notation, which otherwise does not effect the main structure of the proof.

Table 3: The Larger Attack-Defense Game.

	A	B	C	D	E	F
A	$(3,3)$	$(3,2.5)$	$(3,2)$	$(3,1.5)$	$(3,1)$	$(-2,0)$
B	$(2.5,3)$	$(2.5,2.5)$	$(2.5,2)$	$(2.5,1.5)$	$(2.5,1)$	$(0,0)$
C	$(2,3)$	$(2,2.5)$	$(2,2)$	$(2,1.5)$	$(2,1)$	$(0.75,0)$
D	$(1.5,3)$	$(1.5,2.5)$	$(1.5,2)$	$(1.5,1.5)$	$(1.5,1)$	$(1,0)$
E	$(1,3)$	$(1,2.5)$	$(1,2)$	$(1,1.5)$	$(1,1)$	$(1,0)$
F	$(0,-2)$	$(0,0)$	$(0,0.75)$	$(0,1)$	$(0,1)$	$(0,0)$

Proof. **Base Case.** Without loss of generality, assume that all trajectories end in a single state where agents' decisions affect nothing. Such a state can be added without changing the power or utility of any trajectory. At this state all policies are equally valued, so the base case holds trivially.

Inductive Step. Assume that, at any state s' reachable at time $t-1$ from the end, $U_i^{\text{p-Adv}}(\pi_i; \pi_{-i}|s') = U_i(\pi_i; \pi_{-i}|s')$. Our goal is to show this equivalence also holds for states reachable at time t. Expanding out the definition of the p-Adversarial game, we have:

$$\begin{aligned}
&U_i^{\text{p-Adv}}(\pi_i' \quad ; \pi_{-i}|s)\\
=&R_i^{\text{task}}(s, \pi_{-i}) + (1 - \frac{\lambda}{T})\,\mathbb{E}[U_i^{\text{p-Adv}}(\pi_i'; \pi_{-i}|s')]\\
&\qquad + \frac{\lambda}{T}\,\mathbb{E}[U_i^{\text{task}}(\pi_i'; \pi_{-i}|s'_{Adv})]\\
=&R_i^{\text{task}}(s) \qquad + (1 - \frac{\lambda}{T})\,\mathbb{E}[U_i(\pi_i'; \pi_{-i}|s')]\\
&\qquad + \frac{\lambda}{T}\,\mathbb{E}[U_i^{\text{task}}(\pi_i'; \pi_{-i}|s'_{Adv})]
\end{aligned}$$

where $s' \sim T(\pi_i'; \pi_{-i})$, $s'_{Adv} \sim T(\pi_i'; \pi_j^{A^*i}; \pi_{-\{i,j\}})$, and $U_i^{\text{p-Adv}}(\pi|s)$ is the utility of agent i given policies π starting at state s of the p Adversarial game before the adversary has taken control. The first line above follows from the definition of the p adversarial game and the second line follows from the inductive hypothesis. We can continue by expanding out the definitions and rearranging terms:

$$\begin{aligned}
&R_i^{\text{task}}(s) && + (1 - \frac{\lambda}{T})\,\mathbb{E}[U_i^{task}(\pi_i'; \pi_{-i}|s')]\\
&+ \lambda(1 - \frac{\lambda}{T})\,\mathbb{E}[U_i^{\text{power}}(\pi_i'; \pi_{-i}|s')] && + \frac{\lambda}{T}\,\mathbb{E}[U_i^{\text{task}}(\pi_i'; \pi_{-i}|s'_{Adv})]\\
=&R_i^{\text{task}}(s) && + \mathbb{E}[U_i^{\text{task}}(\pi_i'; \pi_{-i}|s')]\\
& && + \lambda(1 - \frac{\lambda}{T})\,\mathbb{E}[U_i^{\text{power}}(\pi_i'; \pi_{-i}|s')]\\
&- \frac{\lambda}{T}\Big(\mathbb{E}[U_i^{\text{task}}(\pi_i'; \pi_{-i}|s')] && - \mathbb{E}[U_i(^{\text{task}}\pi_i'; \pi_{-i}|s'_{Adv})]\Big)\\
=&U_i^{\text{task}}(\pi|s) && + \lambda U_i^{\text{power}}(\pi|s)
\end{aligned}$$

where the final line follows from the definition of task and power utility. Thus, the value of policies in the p-Adversarial game at state s time step t is equivalent to the power-regularized value..

By induction, the equivalence holds for states reachable at any time step. Thus, we have the desired equivalence $U_i^{\text{p-Adv}}(\pi_i; \pi_{-i}|s) = U_i(\pi_i; \pi_{-i}|s)$. □

Given the equivalence between the standard best response in the p-adversarial game and the power-regularized best response in the original game, we can return to our task of proving Theorem 5.3, to show that power regularizing equilibria of G always exist.

Proof. Note that we can construct a tuple of policies π which are a local Nash equilibrium at s in the p-adversarial game at s by standard backwards induction argument – noting first that this can be done in the terminal states, and then noting that it can then be done at each of the prior states backwards by induction.

Consider the policy π_i of an arbitrary player i, by Theorem 5.5, π_i must be a power regularized best response in G at state s. Since we assumed i and s to be arbitrary, this applies to all i and s. Thus the tuple π_i represents a power regularizing equilibrium of G at all states s by definition. □

We have shown that the power regularizing equilibria we seek do, in fact, exist, but moreover, Theorem 5.5 gives some idea about a method to actually obtain them. We can find policies in power-regularizing equilibrium by finding policies in Nash equilibrium in the p-adversarial game of G. This intuition is the motivation behind one of our methods, Sample Based Power Regularization, which we introduce in Section 6.1 and empirically evaluate in Section 7.

6 METHODS

We introduce two methods for power regularization, Sample Based Power Regularization (SBPR) and Power Regularization via Intrinsic Motivation (PRIM). SBPR is inspired by the p-Adversarial Game formulation introduced in Definition 5.4; it injects adversarial data during training by perturbing actions at each step with probability $p = \lambda$. PRIM trains agents directly on the power regularized objective, interpreting the power penalty as intrinsic motivation.

Agents do not share weights, but we train them together offline. Our theory and methods are amenable to be combined with approaches from ad-hoc team play [2, 24] and zero-shot coordination [12, 26], though these domains bring with them their own challenges, so we leave it to future work to generalize this approach.

We train each agent using Proximal Policy Optimization (PPO). The neural networks parameterizing the policy consist of several convolutional layers, fully connected layers, a categorical output head for the actor, and a linear layer value function output head.

6.1 Sample-Based Power Regularization (SBPR)

SBPR directly injects adversarial rollouts into the training data, playing the p-Adversarial Game introduced in Definition 5.4. At the beginning of every rollout, we pick an agent i to be the ego agent and another agent j to be the adversary. At every timestep, with probability p (independent of previous timesteps) we perturb agent j's action adversarially to i. Only agent i receives the rollout to train on. Intuitively, this can be seen as training on the p-Adversarial Game at random states s.

SBPR has the advantage of simplicity but may not regularize for power successfully if we want λ to be very small. This is because we won't see enough deviation examples per batch, so the gradient signal is very high variance, which is reflected in our experiments.

Algorithm 1 Sample Based Power Regularization (SBPR)

```
1: procedure TRAINAGENT($\pi_i, \pi_j, \hat{\pi}_j, V_i, p$)
2:     Collect trajectories for agent $i$: at each timestep, always use $\pi_i$, and with probability $p$ use $\hat{\pi}_j$, otherwise use $\pi_j$.
3:     Use PPO or other RL algorithm to update $\pi_i$ and $V_i$.
4: procedure TRAINADVERSARIALAGENT($\pi_i, \hat{\pi}_j, V_i$)
5:     Collect trajectories for adversarial agent $j$: $\hat{\pi}_j$ has one step to act, and $r_i = V_i(s) - r(s, \pi_i(s), \hat{\pi}_j(s)) - \gamma V_i(s')$.
6:     Use PPO or other RL algorithm to update $\hat{\pi}_j$.
7: procedure SBPR($p$)
8:     Initialize $\pi_i, \pi_j, \hat{\pi}_i, \hat{\pi}_j, V_i, V_j$ arbitrarily.
9:     loop
10:        TRAINAGENT($\pi_i, \pi_j, \hat{\pi}_j, p$)
11:        TRAINAGENT($\pi_j, \pi_i, \hat{\pi}_i, p$)
12:        TRAINADVERSARIALAGENT($\pi_j, \hat{\pi}_i, V_j$)
13:        TRAINADVERSARIALAGENT($\pi_i, \hat{\pi}_j, V_i$)
```

6.2 Power Regularization via Intrinsic Motivation (PRIM)

PRIM adds a per-step intrinsic motivation reward term that penalizes power on the agent. Each agent's reward function is

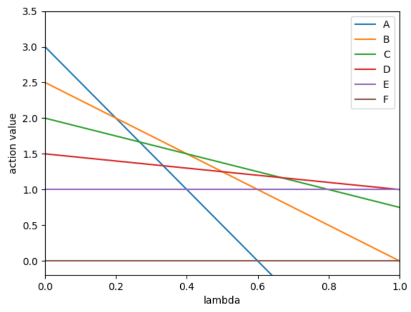
$$R_i^{PRIM}(s, a, \pi) = R_i^{task}(s, a) + \lambda R_i^{power}(s, \pi)$$

which is precisely the power regularization objective of Eq 1. Rather than probabilistically considering the effect of an adversary like SBPR, PRIM considers it at every step but downweights the effect according to λ, which reduces variance.

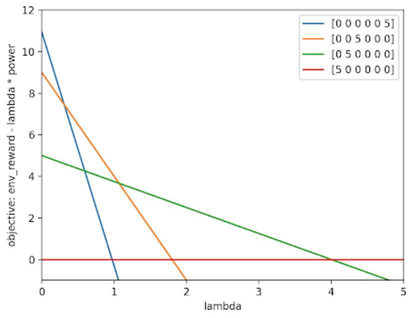
Crucially, finding j's adversarial action for i to compute adversarial power requires counterfactuals from a resettable simulator. In the future one could try to learn the simulator instead.

Algorithm 2 Power Regularization via Intrinsic Motivation (PRIM)

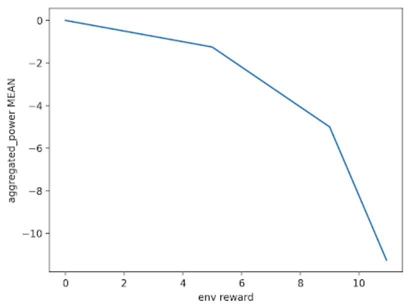
```
1: procedure COMPUTEPOWER($\pi_i, \pi_j, \hat{\pi}_j, V_i$)
2:     $s' \sim T(\cdot|s, \pi_i(s), \pi_j(s))$
3:     $r \leftarrow r(s, \pi_i(s), \pi_j(s)) + \gamma V_i(s')$
4:     $s'_{adv} \sim T(\cdot|s, \pi_i(s), \hat{\pi}_j(s))$
5:     $r_{adv} \leftarrow r(s, \pi_i(s), \hat{\pi}_j(s)) + \gamma V_i(s'_{adv})$
6:     $power \leftarrow r - r_{adv}$
7:     return $power$
8: procedure TRAINAGENT($\pi_i, \pi_j, \hat{\pi}_j, V_i, \lambda$)
9:     Collect trajectories for agent $i$ with reward $r_{task} + \lambda r_{power}$.
10:    Use PPO or other RL algorithm to update $\pi_i$ and $V_i$.
11: procedure TRAINADVERSARIALAGENT($\pi_i, \hat{\pi}_j, V_i$)
12:    Collect trajectories for adversarial agent $j$: $\hat{\pi}_j$ has one step to act, and $r_i = V_i(s) - r(s, \pi_i(s), \hat{\pi}_j(s)) - \gamma V_i(s')$.
13:    Use PPO or other RL algorithm to update $\hat{\pi}_j$.
14: procedure PRIM($\lambda$)
15:    Initialize $\pi_i, \pi_j, \hat{\pi}_i, \hat{\pi}_j, V_i, V_j$ arbitrarily.
16:    loop
17:        $\pi_i, V_i \leftarrow$ TRAINAGENT($\pi_i, \pi_j, \hat{\pi}_j, V_i, \lambda$)
18:        $\pi_j, V_j \leftarrow$ TRAINAGENT($\pi_j, \pi_i, \hat{\pi}_i, V_j, \lambda$)
19:        $\hat{\pi}_i \leftarrow$ TRAINADVERSARIALAGENT($\pi_j, \hat{\pi}_i, V_j$)
20:        $\hat{\pi}_j \leftarrow$ TRAINADVERSARIALAGENT($\pi_i, \hat{\pi}_j, V_i$)
```



(a) Attack-Defense Game Action Payoffs



(b) Coin Division Game Action Payoffs



(c) Coin Division Game Pareto Frontier

Figure 2: Power-regularized objective values achieved by different actions in small environments.

6.3 Optimizations

On their own, both PRIM and SBPR can be slow or unstable, so we introduce a number of optimizations.

Monte Carlo Estimates of Adversarial Power. When computing adversarial power for PRIM, we use the instantiated actions in each rollout rather than the full policies π, effectively a Monte Carlo estimate. This is because PPO looks at the advantages of actions across the batch, so we need to be able to tell which actions incur more or less power. Policies don't give us this information because they are constant throughout a batch; we get no differential information. As a bonus, using actions gives a speedup by a factor of the joint co-player action space.

Learning the Adversary. Both PRIM and SBPR require finding the co-player j's action that minimizes the ego agent i's reward. For environments with large or continuous action spaces, conducting an exhaustive search may be intractable, so we learn the reward-minimizing action: the adversarial co-player j for ego agent i is trained to minimize i's return:$U^i_{j,adv}(s, a_j) = -R_i(s, a) - \mathbb{E}[U_i(T(s, a))]$ where $a = \{a_j, \pi_{-j}(s)\}$. Each agent must maintain an adversarial model of each co-player.

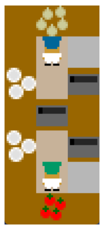
Using the Value Function to Approximate Return from Rollouts. Both PRIM and SBPR require computing the value of states after an adversarial co-player has acted. The benefit of this trick is two-fold: one, it reduces variance because rollouts can be extremely noisy, especially in the beginning of training, and two, it speeds up runtime significantly.

Domain Randomization (DR). DR is useful for speeding up and stabilizing convergence in both methods. Overcooked is a highly sequential environment, requiring a long string of actions to receive a reward, so it is helpful to train starting from random states and learn the optimal policy backwards. Furthermore, crucially for PRIM, DR enables accurate value estimates of states that are off-policy and thus normally not visited. This allows agents to learn how to recover from adversarial deviations and update their value estimates of such states accordingly.

Normalization for the Adversary. The adversary's objective is highly dependent on the starting state because it only gets to act for one timestep, thus the value is high variance which is only worsened by DR. We reduce variance by normalizing the adversary's reward by subtracting the value estimate of the starting state:

$$U^i_{j,adv}(s, a_j) = U_i(s) - R_i(s, a) - \mathbb{E}[U_i(T(s, a))]$$

where $a = \{a_j, \pi_{-j}(s)\}$.


(a) Starting State

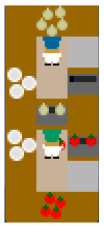

(b) High Power Timestep

Figure 3: Overcooked Close-Pot-Far-Pot. Agents can use the shared middle pot or their private pots. Using the middle pot is faster but incurs high power (see (b)) where one agent can mess up the other's work by putting in a wrong ingredient.

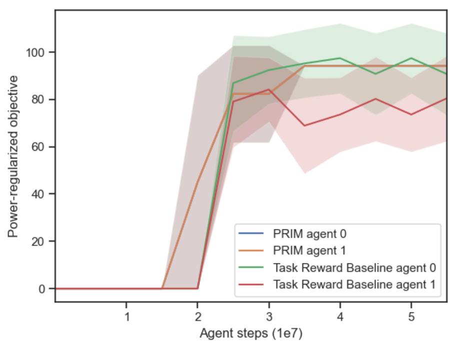
7 EXPERIMENTS

We first validate our methods in small environments where we can compute the optimal actions and then move to larger environments.

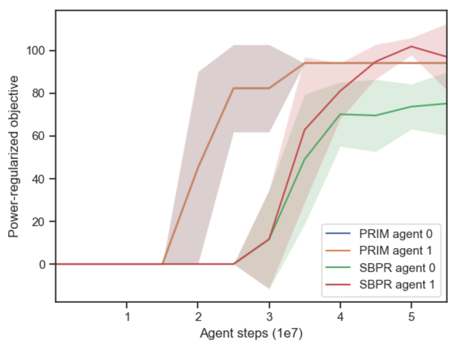
7.1 Small Environments

We evaluate in the larger version of the Attack Defense Game (payoff matrix given in Table 3 and power-regularized objective values per action in Figure 2a) and another environment called the Coin Division game. There are four agents, one "divider" agent (P0), and three "accepter" agents (P1, P2, and P3). There are six bins with the following assignment of agents to bins: ([], [P0], [P0,P1], [P0,P2], [P1,P2], [P1,P2,P3]). The divider agent must allocate five coins amongst the bins. For each bin, the agents assigned to that bin have the option of accepting or rejecting. If everyone accepts, everyone takes home the number of coins assigned to that bin times the number of agents assigned to that bin. If one or more agents reject, the coins assigned to that bin are destroyed. We consider the divider agent's optimal policy and assume that all accepter agents always accept 90% of the time (per bin).

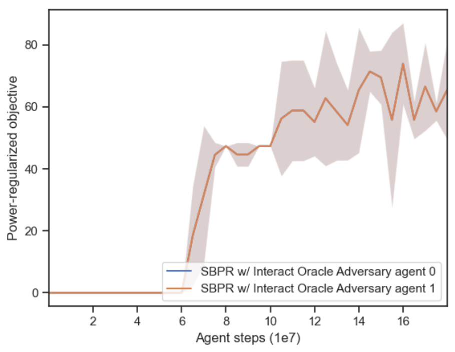
See Figure 2b for the power-regularized objective values of each action (omitting actions which are strictly dominated) and Figure 2c



(a) PRIM vs Baseline (Explosion)



(b) PRIM vs SBPR (Explosion)



(c) PRIM vs SBPR (Explosion)

Figure 4: Comparison of PRIM, SBPR, and Task-Only Baseline in Overcooked Explosion with $\lambda = 0.0001$. In some runs only one agent is visible because the powers incurred are too small to be distinguishable after multiplying by λ. Error bars are standard deviations over 5 trials except for SBPR interact oracle which only has 3 trials.

Table 4: Experimental Results in Explosion Environment. Error values are standard deviations over 5 trials except for SBPR interact oracle which only has 3 trials.

Name	*Task reward*	*Power on Agent 0*	*Power on Agent 1*	*PR Objective Agent 0*	*PR Objective Agent 1*
Task-only baseline	104.7 ± 0.4	139949.5 ± 171647.1	242363.6 ± 182484.0	90.7 ± 17.3	80.5 ± 18.1
SBPR	105.0 ± 0.0	297351.0 ± 148585.0	78369.8 ± 156414.3	75.2 ± 14.8	97.1 ± 15.7
SBPR INTERACT adversary	65.5 ± 16.2	174.0 ± 176.0	217.8 ± 212.6	65.5 ± 16.2	65.5 ± 16.2
PRIM	94.2 ± 0.0	138.4 ± 2.7	132.3 ± 40.5	94.2 ± 2.2	94.2 ± 0.0

for the corresponding Pareo frontier. Both PRIM and SBPR achieve the optimal actions for values of λ sampled in the range 0 to 1.

7.2 Overcooked: Close-Pot-Far-Pot

We evaluate both SBPR and PRIM in Overcooked, a 2 player grid-world game where the objective is to prepare and deliver soups according to given recipes. Recipes may call for two types of ingredients, tomatoes and onions. Agents must collect and place all ingredients in a pot one at a time, cook the soup, grab a plate, place the finished soup onto the plate, and finally deliver the soup.

The action space is {`N`, `E`, `S`, `W`, `STAY`, `INTERACT`}. Depending on where the agent is facing, `INTERACT` can mean pick up an ingredient, place an ingredient into a pot, start cooking, pick up a dish, place soup onto the dish, or deliver a soup. It's impossible to remove ingredients from a pot once they are placed.

We design a layout "Close-Pot-Far-Pot" with two recipes, 3 tomatoes or 3 onions, each giving R reward. The top agent can only access onions and the bottom agent can only access tomatoes. Each agent can access two pots, one shared in the center and the other is private, inaccessible to the other agent, but further. The agents share a reward function and a trajectory is T steps.

In our experiments we set $T = 105$ and $R = 20$. An assembly line (strategy 1) using the middle pot can produce 7 soups, one agent independently using the middle pot and the other using their private pot can produce 9 soups (strategy 2), and both agents independently using their private pots can produce 8 soups (strategy 3).

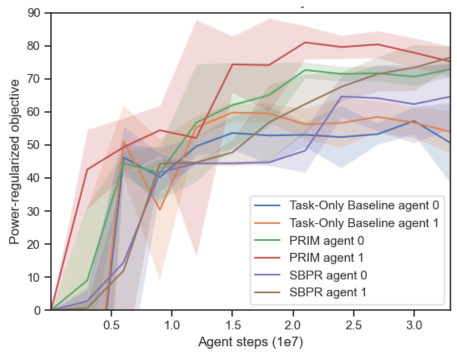
Strategy 2 maximizes task reward but incurs high adversarial power: as shown in Figure 3b, the tomato agent can mess up the onion agent's soup by putting in a wrong ingredient, leading to the onion agent making fewer soups. The state depicted in Figure 3b is on-policy since the tomato agent must move up before turning right to face its private pot to place its tomato there.

We compare the performance of our methods to the task-only baseline. We compute ground truth power through an exhaustive search for the return-minimizing action and conduct full rollouts to evaluate resulting states. This is extremely slow so we only calculate it once every several hundred training iterations. In general rollouts are high variance so multiple trials should be performed, but since our agents converge towards deterministic policies in our environments, we simply determinize the policies when rolling out.

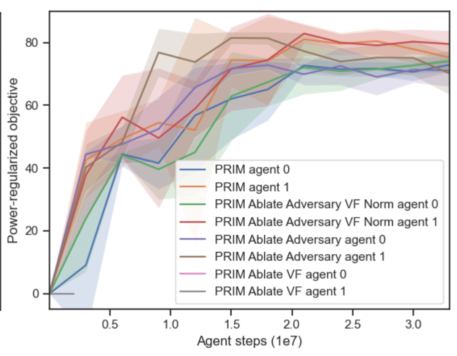
For $\lambda = 0.25$, Figure 5a shows that PRIM outperforms the baseline of optimizing for just the task reward. PRIM also performs better than SBPR for one agent due to its inherently lower variance training data which makes the learning problem easier.

Next we ran a series of ablation experiments to better understand PRIM, shown in Figure 5b. Ablating the learned adversary and instead conducting an exhaustive search over the action space did not make much difference on the objective value achieved. This is expected; the goal of learning the adversary is simply to speed up the power computation: rather than iterating over the action space, we pay a "fixed cost" to train and query the adversary. This is is necessary in environments with large action spaces.

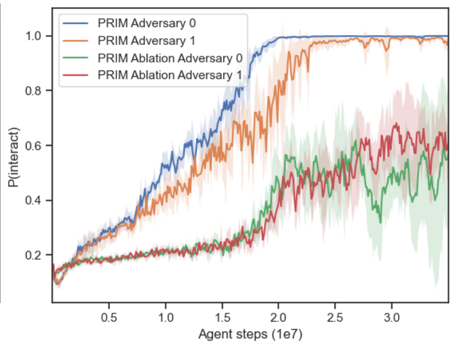
Ablating normalization for the adversary's objective did not significantly change the objective value achieved, but it did hurt the adversary's convergence. Figure 5c depicts the poor convergence for the state in Figure 3b where the optimal action is `INTERACT`.



(a) PRIM vs SBPR vs Task-Only Baseline



(b) PRIM Ablations



(c) Adversary Policy Convergence

Figure 5: Experimental Results in Overcooked Close-Pot-Far-Pot. Error bars are standard deviations over 5 trials except for PRIM ablate VF which only has 3 trials.

Table 5: End-of-training metrics in Overcooked Close-Pot-Far-Pot. Error values are standard deviations over 5 trials except for PRIM ablate VF which only has 3 trials.

Name	*Task reward*	*Power on Agent 0*	*Power on Agent 1*	*PR Objective Agent 0*	*PR Objective Agent 1*
Task-only baseline	104.9 ± 0.2	217.1 ± 47.9	203.8 ± 26.1	50.6 ± 12.1	53.9 ± 6.6
SBPR	94.2 ± 0.0	118.4 ± 40.3	71.1 ± 13.8	64.6 ± 10.1	76.4 ± 3.4
PRIM	94.4 ± 0.1	86.0 ± 9.2	76.6 ± 18.6	72.9 ± 2.2	75.2 ± 4.7
PRIM ablate adversary norm	94.3 ± 0.1	80.8 ± 14.5	59.1 ± 17.2	74.0 ± 3.5	79.5 ± 4.3
PRIM ablate adversary	94.5 ± 0.2	93.6 ± 20.3	97.4 ± 27.1	71.1 ± 5.1	70.1 ± 6.6
PRIM ablate VF	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0

Finally we ablated the use of value function to approximate the return from a rollout. We ran this experiment for the same amount of time as the other experiments, but it was so slow it was only able to finish 2e6 agent steps and achieved 0 on the task reward. We summarize the results of our experiments in Table 5.

7.3 Overcooked: Explosion

In the Close-Pot-Far-Pot layout, an adversarial deviation does not have large consequences, but power regularization may be more useful in high stakes events. We create a variant of Close-Pot-Far-Pot called Explosion where we interpret the ingredients as chemicals, the pots as test tubes, and the recipes as chemical formulas. If unlike chemicals are mixed together, a dangerous chemical reaction causes an explosion which incurs an immediate penalty of $P = -100,000$.

Figure 4a compares the task reward only baseline to PRIM with $\lambda = 0.0001$. Note that the blue line is hidden beneath the orange line. PRIM converges to very low variance while the baseline has high variance. This is due in large part to the fact that agents 0 and 1 may switch roles in who uses the shared pot so either agent may incur the large power penalty.

Now we examine SBPR's performance (see Figure 4b). We expected SBPR to fail since the probability of a deviation $p = 0.0001$ is so low yet the explosion penalty $P = -100,000$ is so high, but the observed performance was better than expected. However, a significant amount of hyperparameter tuning was necessary: we adjusted the PPO clip param and maximum grad norm down to 0.1 and lengthened the entropy schedule. Depending on the particular hyperparameter values, the agents would either fail to optimize for power at all or would converge on an assembly line that avoids the explosion risk (but is suboptimal to PRIM's solution).

As shown in Figure 4c, SBPR relies on the adversary not yet converging at the beginning because this allows the agents to solve enough of the exploration problem before consistently incurring the penalty. Replacing the adversary with an agent that always plays INTERACT (an *interact oracle*) causes SBPR to fail.

We summarize the Explosion results in Table 4. PRIM is the only method that avoids incurring catastrophically high power at the cost of a bit of task reward.

8 CONCLUSION

We defined a notion of power amenable to optimization and showed that equilibria always exist when agents regularize for power. Next, we presented two algorithms, Sample Based Power Regularization (SBPR) and Power Regularization via Intrinsic Motivation (PRIM). We validate our methods in a series of small environments and in two variants of Overcooked, showing that both methods guide agents toward lower power behavior. SBPR is simpler but PRIM is better able to handle very low values of λ.

There are many avenues for future work, including exploring different definitions of power (empirically and philosophically) and modeling multiple timestep deviations. Our theoretical results hold for general-sum games but we have not explored general-sum games empirically.

ACKNOWLEDGMENTS

We are grateful for insightful conversations from the members of the Center for Human-Compatible AI, including Micah Carroll, Niklas Lauffer, Adam Gleave, Daniel Filan, Lawrence Chan, and Sam Toyer and funding of this work as a gift from the Berkeley Existential Risk Intuitive.

REFERENCES

[1] Natasha Alechina, Joseph Y Halpern, and Brian Logan. 2020. Causality, responsibility and blame in team plans. *arXiv preprint arXiv:2005.10297* (2020).

[2] Samuel Barrett, Peter Stone, and Sarit Kraus. 2011. Empirical evaluation of ad hoc teamwork in the pursuit domain. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*. 567–574.

[3] Micah Carroll, Rohin Shah, Mark K Ho, Tom Griffiths, Sanjit Seshia, Pieter Abbeel, and Anca Dragan. 2019. On the utility of learning about humans for human-ai coordination. *Advances in neural information processing systems* 32 (2019).

[4] Hana Chockler and Joseph Y Halpern. 2004. Responsibility and blame: A structural-model approach. *Journal of Artificial Intelligence Research* 22 (2004), 93–115.

[5] Virginia Dignum and Frank Dignum. 2006. Coordinating tasks in agent organizations. In *International Workshop on Coordination, Organizations, Institutions, and Norms in Agent Systems*. Springer, 32–47.

[6] Jakob N Foerster, Richard Y Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch. 2017. Learning with opponent-learning awareness. *arXiv preprint arXiv:1709.04326* (2017).

[7] Meir Friedenberg and Joseph Y Halpern. 2019. Blameworthiness in multi-agent settings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 525–532.

[8] Tobias Gerstenberg, Joseph Y Halpern, and Joshua B Tenenbaum. 2015. Responsibility judgments in voting scenarios.. In *CogSci*.

[9] Adam Gleave, Michael Dennis, Cody Wild, Neel Kant, Sergey Levine, and Stuart Russell. 2019. Adversarial policies: Attacking deep reinforcement learning. *arXiv preprint arXiv:1905.10615* (2019).

[10] Davide Grossi, Frank Dignum, Virginia Dignum, Mehdi Dastani, and Làmber Royakkers. 2006. Structural aspects of the evaluation of agent organizations. In *International Workshop on Coordination, Organizations, Institutions, and Norms in Agent Systems*. Springer, 3–18.

[11] Joseph Halpern and Max Kleiman-Weiner. 2018. Towards formal definitions of blameworthiness, intention, and moral responsibility. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[12] Hengyuan Hu, Adam Lerer, Alex Peysakhovich, and Jakob Foerster. 2020. "Other-Play" for Zero-Shot Coordination. In *International Conference on Machine Learning*. PMLR, 4399–4410.

[13] Sandy Huang, Nicolas Papernot, Ian Goodfellow, Yan Duan, and Pieter Abbeel. 2017. Adversarial attacks on neural network policies. *arXiv preprint arXiv:1702.02284* (2017).

[14] Edward Hughes, Joel Z Leibo, Matthew Phillips, Karl Tuyls, Edgar Dueñez-Guzman, Antonio García Castañeda, Iain Dunning, Tina Zhu, Kevin McKee, Raphael Koster, et al. 2018. Inequity aversion improves cooperation in intertemporal social dilemmas. *Advances in neural information processing systems* 31 (2018).

[15] Natasha Jaques, Angeliki Lazaridou, Edward Hughes, Caglar Gulcehre, Pedro Ortega, DJ Strouse, Joel Z Leibo, and Nando De Freitas. 2019. Social influence as intrinsic motivation for multi-agent deep reinforcement learning. In *International conference on machine learning*. PMLR, 3040–3049.

[16] Jernej Kos and Dawn Song. 2017. Delving into adversarial attacks on deep policies. *arXiv preprint arXiv:1705.06452* (2017).

[17] Joel Z Leibo, Vinicius Zambaldi, Marc Lanctot, Janusz Marecki, and Thore Graepel. 2017. Multi-agent reinforcement learning in sequential social dilemmas. *arXiv preprint arXiv:1702.03037* (2017).

[18] Yen-Chen Lin, Zhang-Wei Hong, Yuan-Hong Liao, Meng-Li Shih, Ming-Yu Liu, and Min Sun. 2017. Tactics of adversarial attack on deep reinforcement learning agents. *arXiv preprint arXiv:1703.06748* (2017).

[19] Michael L Littman. 1994. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*. Elsevier, 157–163.

[20] Michael L Littman et al. 2001. Friend-or-foe Q-learning in general-sum games. In *ICML*, Vol. 1. 322–328.

[21] Steven Lukes. 2021. *Power: A radical view*. Bloomsbury Publishing.

[22] Michael Mann. 2012. *The sources of social power: volume 1, a history of power from the beginning to AD 1760*. Vol. 1. Cambridge university press.

[23] Jack Serrino, Max Kleiman-Weiner, David C Parkes, and Josh Tenenbaum. 2019. Finding friend and foe in multi-agent games. *Advances in Neural Information Processing Systems* 32 (2019).

[24] Peter Stone, Gal A. Kaminka, Sarit Kraus, and Jeffrey S. Rosenschein. 2010. Ad Hoc Autonomous Agent Teams: Collaboration without Pre-Coordination. In *Proceedings of the Twenty-Fourth Conference on Artificial Intelligence*.

[25] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* (2013).

[26] Johannes Treutlein, Michael Dennis, Caspar Oesterheld, and Jakob Foerster. 2021. A new formalism, method and open issues for zero-shot coordination. In *International Conference on Machine Learning*. PMLR, 10413–10423.

[27] Alex Turner, Neale Ratzlaff, and Prasad Tadepalli. 2020. Avoiding side effects in complex environments. *Advances in Neural Information Processing Systems* 33 (2020), 21406–21415.

[28] Alexander Matt Turner, Dylan Hadfield-Menell, and Prasad Tadepalli. 2020. Conservative agency via attainable utility preservation. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*. 385–391.

[29] Alexander Matt Turner, Logan Smith, Rohin Shah, Andrew Critch, and Prasad Tadepalli. 2019. Optimal Policies Tend to Seek Power. *arXiv preprint arXiv:1912.01683* (2019).