Inverse Reinforcement Learning From Like-Minded Teachers

Ritesh Noothigattu, [1*]**, Tom Yan,**[1*] **Ariel D. Procaccia**[2]
[1] Carnegie Mellon University
[2] Harvard University
{riteshn, tyyan}@cmu.edu, arielpro@seas.harvard.edu

Abstract

We study the problem of learning a policy in a Markov decision process (MDP) based on observations of the actions taken by multiple teachers. We assume that the teachers are like-minded in that their (linear) reward functions — while different from each other — are random perturbations of an underlying reward function. Under this assumption, we demonstrate that inverse reinforcement learning algorithms that satisfy a certain property — that of *matching feature expectations* — yield policies that are approximately optimal with respect to the underlying reward function, and that no algorithm can do better in the worst case. We also show how to efficiently recover the optimal policy when the MDP has one state — a setting that is akin to multi-armed bandits.

Introduction

A *Markov decision process (MDP)* is a formal specification of a sequential decision making environment, which consists of a set of states, a set of actions, a reward function, and a stochastic transition function. *Reinforcement learning (RL)* deals with learning a *policy* in an MDP — which specifies a possibly randomized action that is taken in each state — to maximize cumulative reward.

RL has long history in AI (Sutton and Barto 1998; Kaelbling, Littman, and Moore 1996), as well as in many other disciplines. But in recent years, interest in the area has exploded, in part due to breakthroughs in game playing (Mnih et al. 2015; Silver et al. 2016) and fast-growing applications to robotics (Kober, Bagnell, and Peters 2013). It is safe to say that, nowadays, RL is widely considered to be one of the basic building blocks in the construction of intelligent agents.

While most work in the area focuses on maximizing a given reward function, some settings require the AI system to emulate the behavior of an expert or teacher (Ng and Russell 2000; Abbeel and Ng 2004) — this is known as *inverse reinforcement learning (IRL)*. The idea is to observe an agent executing a policy in an MDP, where everything is known to the learner except the reward function, and extract a reward function that is most likely to be the one being optimized by the agent. Using this reward function — and knowledge of the other components of the MDP — the agent can easily compute an optimal policy to follow.

Our point of departure is that we are interested in IRL from multiple agents rather than a single agent. Specifically, we observe n different agents executing policies that are optimal for their individual reward functions. Our approach is to aggregate these observations into a single policy, by applying an inverse reinforcement learning algorithm to the set of all observations.

However, if individual agents have wildly divergent reward functions then the aggregate policy may not represent coherent behavior. In addition, to formally reason about the quality of the optimal policy, we need to relate it to some notion of ground truth. For these reasons, we assume that the agents are *like-minded*, in that individual reward functions are nothing but noisy versions of an underlying reward function.

In summary, our research challenge is this:

> *Given observations from policies that are optimal with respect to different reward functions, each of which is a perturbation of an underlying reward function, identify IRL algorithms that can recover a good policy with respect to the underlying reward function.*

We believe that this problem is both natural and general. To further motivate it, though, let us briefly instantiate it in the context of beneficial AI. One of the prominent approaches in this area is to align the values of the AI system with the values of a human through IRL (Russell, Dewey, and Tegmark 2015; Hadfield-Menell et al. 2016). Our extension to multiple agents would allow the alignment of the system with the values of *society*.

A compelling aspect of this instantiation is that, if we think of the underlying reward function as embodying a common set of moral propositions, then our technical assumption of like-minded agents can be justified through the *linguistic analogy*, originally introduced by Rawls (1971). It draws on the work of Chomsky (1965), who argued that competent speakers have a set of grammatical principles in mind, but their linguistic behavior is hampered by "grammatically irrelevant conditions such as memory limitations, distractions, shifts of attention and interest, and errors." Analogously, Rawls claimed, humans have moral rules — a common "moral grammar" — in our minds, but, due to

[*] Equal contribution

various limitations, our moral behavior is only an approximation thereof. Interestingly, this theory lends itself to empirical experimentation, and, indeed, it has been validated through work in moral psychology (Mikhail 2011).

Our Model and Results. We start from a common IRL setup: each reward function is associated with a weight vector $\mathbf{w}$, such that the reward for taking a given action in a given state is the dot product of the weight vector and the feature vector of that state-action pair. The twist is that there is an underlying reward function represented by a weight vector $\mathbf{w}^\star$, and each of the agents is associated with a weight vector $\mathbf{w}_i$, which induces an optimal policy π_i. We observe a trajectory from each π_i.

In Section , we focus on competing with a uniform mixture over the optimal policies of the agents, $\pi_1, \ldots, \pi_n$ (for reasons that we explicate momentarily). We can do this because the observed trajectories are "similar" to the uniform mixture, in the sense that their feature vectors — the discounted frequencies of the features associated with the observed state-action pairs — are close to that of the uniform mixture policy. Therefore, due to the linearity of the reward function, any policy whose feature expectations approximately match those of the observed trajectories must be close to the uniform mixture with respect to $\mathbf{w}^\star$. We formalize this idea in Theorem 0.2, which gives a lower bound on the number of agents and length of observed trajectories such that any policy that $\epsilon/3$-matches feature expectations is ϵ-close to the uniform mixture. Furthermore, we identify two well-known IRL algorithms, Apprenticeship Learning (Abbeel and Ng 2004) and Max Entropy (Ziebart et al. 2008), which indeed output policies that match the feature expectations of the observed trajectories, and therefore enjoy the guarantees provided by this theorem.

Needless to say, competing with the uniform mixture is only useful insofar as this benchmark exhibits "good" performance. We show that this is indeed the case in Section , assuming (as stated earlier) that each weight vector $\mathbf{w}_i$ is a noisy perturbation of $\mathbf{w}^\star$. Specifically, we first establish that, under relatively weak assumptions on the noise, it is possible to bound the difference between the reward of the uniform mixture and that of the optimal policy (Theorem 0.3). More surprisingly, Theorem 0.5 asserts that in the worst case it is impossible to outperform the uniform mixture, by constructing an MDP where the optimal policy cannot be identified — even if we had an infinite number of agents and infinitely long trajectories! Putting all of these results together, we conclude that directly running an IRL algorithm that matches feature expectations on the observed trajectories is a sensible approach to our problem.

Nevertheless, it is natural to ask whether it is possible to outperform the uniform mixture in typical instances. In Section we show that this is indeed the case; in fact, we are able to recover the optimal policy whenever it is identifiable, albeit under stringent assumptions — most importantly, that the MDP has only one state. This leads to a challenge that we call the *inverse multi-armed bandit problem*. To the best of our knowledge, this problem is novel; its study contributes to the (relatively limited) understanding of scenarios where it is possible to outperform teacher demonstrations.

Related work. The most closely related work deals with IRL when the observations come from an agent who acts according to multiple *intentions*, each associated with a different reward function (Babeş-Vroman et al. 2011; Choi and Kim 2012). The main challenge stems from the need to cluster the observations — the observations in each cluster are treated as originating from the same policy (or intention). By contrast, clustering is a nonissue in our framework. Moreover, our assumption that each $\mathbf{w}_i$ is a noisy perturbation of $\mathbf{w}^\star$ allows us to provide theoretical guarantees.

Further afield, there is a body of work on robust RL and IRL under reward uncertainty (Givan, Leach, and Dean 2000; Regan and Boutilier 2009, 2010), noisy rewards (Zheng, Liu, and Ni 2014), and corrupted rewards (Everitt et al. 2017). Of these papers the closest to ours is that of Zheng, Liu, and Ni (2014), who design robust IRL algorithms under *sparse* noise, in the sense that only a small fraction of the observations are anomalous; they do not provide theoretical guarantees. Our setting is quite different, as very few observations would typically be associated with a near-perfect policy.

MDP Terminology

We assume the environment is modeled as an MDP $\{S, A, T, \gamma, D\}$ with an unknown reward function. S is a finite set of states; A is a finite set of actions; $T(s, a, s')$ is the state transition probability of reaching state s' from state s when action a is taken; $\gamma \in [0, 1)$ is the discount factor; and D the initial-state distribution, from which the start state s_0 is drawn for every trajectory.

As is standard in the literature (Abbeel and Ng 2004), we assume that there is a function $\phi : S \times A \to \mathbb{R}^d$ that maps state-action pairs to their real-valued features. We also overload notation, and say that the feature vector of a trajectory $\tau = \{(s_0, a_0), (s_1, a_1), \ldots, (s_L, a_L)\}$ is defined as $\phi(\tau) = \sum_{t=0}^{L} \gamma^t \phi(s_t, a_t)$.

We make the standard assumption that the immediate reward of executing action a from state s is linear in the features of the state-action pair, i.e. $r^{\mathbf{w}}(s, a) = \mathbf{w}^\intercal \phi(s, a)$. This has a natural interpretation: ϕ represents the different factors, and $\mathbf{w}$ weighs them in varying degrees. Note that we assume that ϕ is a sufficiently rich feature extractor that can capture complex state-action-reward behavior; for instance, if one is to view $r^w(s, a)$ as a neural network, ϕ corresponds to all but the last layer of the network.

Let μ denote the feature expectation of policy π, that is, $\mu(\pi) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t \phi(s_t, a_t) | \pi]$, where π defines the action a_t taken from state s_t, and the expectation is taken over the transition probabilities $T(s_t, a_t, s_{t+1})$. Therefore, since $R^{\mathbf{w}}(\pi) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r^{\mathbf{w}}(s_t, a_t) \middle| \pi\right]$, the cumulative reward of a policy π under weight $\mathbf{w}$ can be rewritten as:

$$R^{\mathbf{w}}(\pi) = \mathbf{w}^\intercal \cdot \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \phi(s_t, a) \middle| \pi\right] = \mathbf{w}^\intercal \mu(\pi)$$

Let $P_\pi(s,t)$ denote the probability of getting to state s at time t under policy π. Then, the cumulative reward $R^{\mathbf{w}}$ is

$$R^{\mathbf{w}}(\pi) = \sum_{t=0}^{\infty} \gamma^t \sum_{s \in S} P_\pi(s,t) r^{\mathbf{w}}(s, \pi(s)).$$

Approximating the Uniform Mixture

We consider an environment with n agents $N = \{1, \ldots, n\}$. Furthermore, the reward function of each agent $i \in N$ is associated with a weight vector $\mathbf{w}_i$, and, therefore, with a reward function $r^{\mathbf{w}_i}$. This determines the optimal policy π_i executed by agent i, from which we observe the trajectory τ_i, which consists of L steps. We observe such a trajectory for each $i \in N$, giving us trajectories $\{\tau_1, ..., \tau_n\}$.

As we discussed in Section , we assume that the reward function associated with each agent is a noisy version of an underlying reward function. Specifically, we assume that there exists a ground truth weight vector $\mathbf{w}^\star$, and for each agent $i \in N$ we let $\mathbf{w}_i = \mathbf{w}^\star + \boldsymbol{\eta}_i$, where $\boldsymbol{\eta}_i$ is the corresponding noise vector; we assume throughout that $\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_n$ are i.i.d. Following Abbeel and Ng (2004), we also assume in some of our results (when stated explicitly) that $\|\mathbf{w}^\star\|_2 \leq 1$ and $\|\phi(s,a)\|_\infty \leq 1$.

Let us denote by π^u the *uniform mixture* over the policies $\pi_1, \ldots, \pi_n$, that is, the (randomized) policy that, in each trajectory, selects one of these policies uniformly at random and executes it throughout the trajectory.

Our goal in this section is to "approximate" the uniform mixture (and we will justify this choice in subsequent sections). To do so, we focus on IRL algorithms that "match feature expectations." Informally, the property of interest is that the feature expectations of the policy match the (discounted) feature vectors of observed trajectories. This idea is already present in the IRL literature, but it is helpful to define it formally, as it allows us to identify specific IRL algorithms that work well in our setting.

Definition 0.1. Given n trajectories $\tau_1, ..., \tau_n$, a (possibly randomized) policy π *ϵ-matches their feature expectations* if and only if $\|\mu(\pi) - \frac{1}{n}\sum_{i=1}^n \phi(\tau_i)\|_2 \leq \epsilon$.

In a nutshell, due to the linearity of the reward function, two policies that have the same feature expectations have the same reward. Therefore, if the observed trajectories closely mimic the feature expectations of π_u, and a policy $\tilde{\pi}$ matches the feature expectations of the observed trajectories, then the reward of $\tilde{\pi}$ would be almost identical to that of π^u. This is formalized in the following theorem, whose proof is relegated to the full version of the paper.

Theorem 0.2. *Assume that $\|\phi(s,a)\|_\infty \leq 1$ for all $s \in S, a \in A$. Let $\mathbf{w}^\star$ such that $\|\mathbf{w}^\star\|_2 \leq 1$, fix any $\mathbf{w}_1, \ldots, \mathbf{w}_n$, and, for all $i \in N$, let τ_i be a trajectory of length L sampled by executing π_i. Let $\tilde{\pi}$ be a policy that $\epsilon/3-$matches the feature expectation of these trajectories. If*

$$n \geq \frac{72 \ln\left(\frac{2}{\delta}\right) d}{\epsilon^2 (1-\gamma)^2} \quad \text{and} \quad L \geq \log_{1/\gamma} \frac{3\sqrt{d}}{(1-\gamma)\epsilon}$$

then, with probability at least $1-\delta$, it holds that $\left|R^{\mathbf{w}^\star}(\tilde{\pi}) - R^{\mathbf{w}^\star}(\pi^u)\right| \leq \epsilon$.

Note that the required number of agents n may be significant; fortunately, we can expect access to data from many agents in applications of interest. For example, Noothigattu et al. (2018) built a system that decides ethical dilemmas based on data collected from 1.3 million people.

To apply Theorem 0.2, we need to use IRL algorithms that match feature expectations. We have identified two algorithms that satisfy this property: the *Apprenticeship Learning* algorithm of Abbeel and Ng (2004), and the *Max Entropy* algorithm of Ziebart et al. (2008). For completeness we present these algorithms, and formally state their feature-matching guarantees, in the full version of the paper.

How Good is the Uniform Mixture?

In Section we showed that it is possible to (essentially) match the performance of the uniform mixture with respect to the ground truth reward function. In this section we justify the idea of competing with the uniform mixture in two ways: first, we show that the uniform mixture approximates the optimal policy under certain assumptions on the noise, and, second, we prove that in the worst case it is actually impossible to outperform the uniform mixture.

The Uniform Mixture Approximates the Optimal Policy

Recall that for all $i \in N$, $\mathbf{w}_i = \mathbf{w}^\star + \boldsymbol{\eta}_i$. It is clear that without imposing some structure on the noise vectors $\boldsymbol{\eta}_i$, no algorithm would be able to recover a policy that does well with respect to $\mathbf{w}^\star$.

Assumptions. Let us assume, then, that the noise vectors $\boldsymbol{\eta}_i$ are such that the η_{ik} are independent and each η_{ik}^2 is sub-exponential. Formally, a random variable X is *sub-exponential* if there are non-negative parameters (ν, b) such that $\mathbb{E}\left[\exp\left(\lambda(X - \mathbb{E}[X])\right)\right] \leq \exp\left(\nu^2\lambda^2/2\right)$ for all $|\lambda| < 1/b$. This flexible definition simply means that the moment generating function of the random variable X is bounded by that of a Gaussian in a neighborhood of 0. Note that if a random variable is sub-Gaussian, then its square is sub-exponential. Hence, our assumption is strictly weaker than assuming that each η_{ik} is sub-Gaussian.

Despite our assumption about the noise, it is *a priori* unclear that the uniform mixture would do well. The challenge is that the noise operates on the coordinates of the individual weight vectors, which in turn determine individual rewards, but, at first glance, it seems plausible that relatively small perturbations of rewards would lead to severely suboptimal policies. Our result shows that this is not the case: π^u is approximately optimal with respect to $R^{\mathbf{w}^\star}$, in expectation.

Theorem 0.3. *Assume that $\|\phi(s,a)\|_\infty \leq 1$ for all $s \in S, a \in A$. Let $\mathbf{w}^\star$ such that $\|\mathbf{w}^\star\|_2 \leq 1$, and suppose that $\mathbf{w}_1, ..., \mathbf{w}_n$ are drawn from i.i.d. noise around $\mathbf{w}^\star$, i.e., $\mathbf{w}_i = \mathbf{w}^\star + \boldsymbol{\eta}_i$, where each of its coordinates is such that η_{ik}^2 is an independent sub-exponential random variable with parameters (ν, b). Then*

$$\mathbb{E}[R^{\mathbf{w}^\star}(\pi^u)] \geq R^{\mathbf{w}^\star}(\pi^\star) - O\left(d\sqrt{u} + \nu\sqrt{\frac{d}{u}} + \frac{b}{\sqrt{u}}\right),$$

where $u = \frac{1}{d}\sum_{k=1}^{d}\mathbb{E}\left[\eta_{ik}^2\right]$*, and the expectation is taken over the noise.*

The exact expression defining the gap between $\mathbb{E}[R^{\mathbf{w}^\star}(\pi^u)]$ and $R^{\mathbf{w}^\star}(\pi^\star)$ can be found in the proof of Theorem 0.3, which appears in the full version of the paper; we give the asymptotic expression in the theorem's statement because it is easier to interpret. As one might expect, this gap increases as ν or b is increased (and, in a linear fashion). This is intuitive because a smaller ν or b imposes a strictly stronger assumption on the sub-exponential random variable (and its tails).

Theorem 0.3 shows that the gap depends linearly on the number of features d. An example given in the full version of the paper shows that this upper bound is tight. Nevertheless, the tightness holds in the worst case, and one would expect the practical performance of the uniform mixture to be very good. To corroborate this intuition, we provide (unsurprising) experimental results in the full version of the paper.

It is Impossible to Outperform the Uniform Mixture in the Worst Case

An ostensible weakness of Theorem 0.3 is that even as the number of agents n goes to infinity, the reward of the uniform mixture may not approach that of the optimal policy, that is, there is a persistent gap. The example given in Section shows the gap is not just an artifact of our analysis. This is expected, because the data contains some agents with suboptimal policies π_i, and a uniform mixture over these suboptimal policies must itself be suboptimal.

It is natural to ask, therefore, whether it is generally possible to achieve performance arbitrarily close to $\pi^\star$ (at least in the limit that n goes to infinity). The answer is negative. In fact, we show that — in the spirit of *minimax optimality* (Hodges Jr and Lehmann 1950; Perron and Marchand 2002) — one cannot hope to perform better than π^u itself in the worst case. Intuitively, there exist scenarios where it is impossible to tell good and bad policies apart by looking at the data, which means that the algorithm's performance depends on what can be gleaned from the "average data".

This follows from a surprising[1] result that we think of as "non-identifiability" of the optimal policy. To describe this property, we introduce some more notation. The distribution over the weight vector of each agent i, $\mathbf{w}_i = \mathbf{w}^\star + \boldsymbol{\eta}_i$, in turn induces a distribution over the optimal policy π_i executed by each agent. Denote this distribution by $\mathcal{P}(\mathbf{w}^\star)$.[2] Hence, each agent's optimal policy π_i is just a sample from this distribution $\mathcal{P}(\mathbf{w}^\star)$. In particular, as the number of agents goes to infinity, the empirical distribution of their optimal policies would exactly converge to $\mathcal{P}(\mathbf{w}^\star)$.

Assumptions. For the rest of this section, we make minimal assumptions on the noise vector $\boldsymbol{\eta}_i$. In particular, we merely assume that $\boldsymbol{\eta}_i$ follows a continuous distribution and that each of its coordinates is i.i.d. We are now ready to state our non-identifiability lemma.

[1] At least it was surprising for us — we spent significant effort trying to prove the opposite result!

[2] Note that this distribution does not depend on i itself since the noise $\boldsymbol{\eta}_i$ is i.i.d. across the different agents.

Lemma 0.4 (non-identifiability)**.** *For every continuous distribution $\mathcal{D}$ over $\mathbb{R}$, if η_{ik} is independently sampled from $\mathcal{D}$ for all $i \in N$ and $k \in [d]$, then there exists an MDP and weight vectors $\mathbf{w}_a^\star$, $\mathbf{w}_b^\star$ with optimal policies $\pi_a^\star$, $\pi_b^\star$, respectively, such that $\pi_a^\star \neq \pi_b^\star$ but $\mathcal{P}(\mathbf{w}_a^\star) = \mathcal{P}(\mathbf{w}_b^\star)$.*

Even if we had an infinite number of trajectories in our data, and even if we knew the exact optimal policy played by each player i, this information would amount to knowing $\mathcal{P}(\mathbf{w}^\star)$. Hence, if there exist two weight vectors $\mathbf{w}_a^\star$, $\mathbf{w}_b^\star$ with optimal policies $\pi_a^\star$, $\pi_b^\star$ such that $\pi_a^\star \neq \pi_b^\star$ and $\mathcal{P}(\mathbf{w}_a^\star) = \mathcal{P}(\mathbf{w}_b^\star)$, then we would not be able to identify whether the optimal policy is $\pi_a^\star$ or $\pi_b^\star$ regardless of how much data we had.

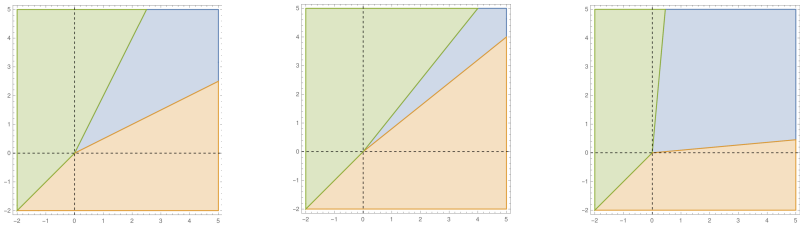
The proof of Lemma 0.4 is relegated to the full version of the paper. Here we provide a proof sketch.

Proof sketch of Lemma 0.4. The intuition for the lemma comes from the construction of an MDP with three possible policies, all of which have probability $1/3$ under $\mathcal{P}(\mathbf{w}^\star)$, even though one is better than the others. This MDP has a single state s, and three actions $\{a, b, c\}$ that lead back to s. Denote the corresponding policies by π_a, π_b, π_c. Let the feature vector be $\phi(s,a) = [0.5, 0.5], \phi(s,b) = [1, -\delta/2], \phi(s,c) = [-\delta/2, 1]$, where $\delta > 0$ is a parameter. Let the ground truth weight vector be $\mathbf{w}^\star = (v_o, v_o)$, where v_o is such that the noised weight vector $\mathbf{w} = \mathbf{w}^\star + \boldsymbol{\eta}$ has probability strictly more than $1/3$ of lying in the first quadrant; an appropriate value of δ always exists for any noise distribution that is continuous and i.i.d. across coordinates s.t the above holds.

Let us look at weight vectors $\mathbf{w}$ for which each of the three policies π_a, π_b and π_c are optimal. π_a is the optimal policy when $\mathbf{w}^\intercal\mu_a > \mathbf{w}^\intercal\mu_b$ and $\mathbf{w}^\intercal\mu_a > \mathbf{w}^\intercal\mu_c$, which is the intersection of the half-spaces $\mathbf{w}^\intercal(-1, 1+\delta) > 0$ and $\mathbf{w}^\intercal(1+\delta, -1) > 0$. Similarly, we can reason about the regions where π_b and π_c are optimal. These regions are illustrated in Figure 1 for different values of δ. Informally, as δ is decreased, the lines separating (π_a, π_c) and (π_a, π_b) move closer to each other (as shown for $\delta = 0.25$), while as δ is increased, these lines move away from each other (as shown for $\delta = 10$). By continuity and symmetry, there exists δ such that the probability of each of the regions (with respect to the random noise) is exactly $1/3$, showing that the MDP has the desired property.

To complete the proof of the lemma, we extend the MDP by adding two more features to the existing two. By setting these new features appropriately (in particular, by cycling the two original features across the arms), we can show that the two weight vectors $\mathbf{w}_a^\star = (v_o, v_o, 0, 0)$ and $\mathbf{w}_b^\star = (0, 0, v_o, v_o)$ lead to $\mathcal{P}(\mathbf{w}_a^\star) = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3}) = \mathcal{P}(\mathbf{w}_b^\star)$, even though their corresponding optimal policies are π_a and π_b, respectively. □

For the next theorem, therefore, we can afford to be "generous:" we will give the algorithm (which is trying to compete with π^u) access to $\mathcal{P}(\mathbf{w}^\star)$, instead of restricting it to sampled trajectories. Formally, the theorem holds for any algorithm that takes a distribution over policies as input, and returns a randomized policy.



$\delta = 1$ $\delta = 0.25$ $\delta = 10$

Figure 1: Regions of each optimal policy for different values of δ. Blue depicts the region where π_a is optimal, orange is where π_b is optimal, and green is where π_c is optimal.

Theorem 0.5. *For every continuous distribution $\mathcal{D}$ over $\mathbb{R}$, if η_{ik} is independently sampled from $\mathcal{D}$ for all $i \in N$ and $k \in [d]$, then there exists an MDP such that for any algorithm $\mathcal{A}$ from distributions over policies to randomized policies, there exists a ground truth weight vector $\mathbf{w}^\star$ such that $R^{\mathbf{w}^\star}(\mathcal{A}(\mathcal{P}(\mathbf{w}^\star))) \leq R^{\mathbf{w}^\star}(\pi^u) < R^{\mathbf{w}^\star}(\pi^\star)$.*

In words, the constructed instance is such that, even given infinite data, no algorithm can outperform the uniform mixture, and, moreover, the reward of the uniform mixture is bounded away from the optimum. The theorem's proof is given in the full version of the paper.

The Inverse Multi-Armed Bandit Problem

In Section , we have seen that it is impossible to outperform the uniform mixture in the worst case, as the optimal policy is not identifiable. However, it is natural to ask when the optimal policy is identifiable and how it may be practically recovered. In this section we give an encouraging answer, albeit in a restricted setting.

Specifically, we focus on the multi-armed bandit problem, which is an MDP with a single state. Note that the non-identifiability result of Lemma 0.4 still holds in this setting, as the example used in its proof is an MDP with a single state. Hence, even in this setting of bandits, it is impossible to outperform the uniform mixture in the worst case. However, we design an algorithm that can guarantee optimal performance when the problem is identifiable, under some additional conditions.

Like the general setting considered earlier, there exists a ground truth weight vector $\mathbf{w}^\star$, and for each agent $i \in N$, $\mathbf{w}_i = \mathbf{w}^\star + \boldsymbol{\eta}_i$. For this section, we assume the noise vector $\boldsymbol{\eta}_i$ to be Gaussian and i.i.d. across agents and coordinates. In particular, $\boldsymbol{\eta}_i \sim \mathcal{N}(0, \sigma^2 I_d)$, and independent across i.

The bandit setting is equivalent to a single-state MDP, and hence the components S, T, γ and D are moot. Instead, there are m arms to pull, denoted by $A = \{1, 2, \ldots, m\}$. Similar to our original feature function ϕ, we now have features $\mathbf{x}_j \in \mathbb{R}^d$ associated with arm j, for each $j \in A$. Although in standard stochastic bandit problems we have a reward sampled from a distribution when we pull an arm, we care only about its mean reward in this section. For weight vector $\mathbf{w}$, the (mean) reward of pulling arm j is given by $r^{\mathbf{w}}(j) = \mathbf{w}^\intercal \mathbf{x}_j$. For each agent i (with weight vector $\mathbf{w}_i$), we assume that we observe the optimal arm being played by this agent, i.e., $\tilde{a}_i = \text{argmax}_{j \in A} \mathbf{w}_i^\intercal \mathbf{x}_j$.

We observe the dataset $\mathcal{D} = \{\tilde{a}_1, \tilde{a}_2, \ldots, \tilde{a}_n\}$ which is the set of optimal arms played by the agents. Define $\mathcal{Q}(\mathbf{w}^\star)$ to be the distribution over optimal arms induced when the ground truth weight vector is $\mathbf{w}^\star$. In particular, ground truth weight vector $\mathbf{w}^\star$ induces a distribution over the noised weight vector of each agent (via $\mathbf{w} = \mathbf{w}^\star + \eta$), which in turn induces a discrete distribution over the optimal arm that would be played, which we call $\mathcal{Q}(\mathbf{w}^\star)$ — analogously to the $\mathcal{P}(\mathbf{w}^\star)$ of Section . Observe that the dataset $\mathcal{D}$ could be rewritten as a distribution over arms, $\tilde{\mathcal{Q}} = (\tilde{\mathcal{Q}}_1, \tilde{\mathcal{Q}}_2, \ldots, \tilde{\mathcal{Q}}_m)$, which is the observed distribution of optimal arms. Moreover, as each agent's optimal arm played is an i.i.d. sample from $\mathcal{Q}(\mathbf{w}^\star)$, the empirical distribution $\tilde{\mathcal{Q}}$ is an unbiased estimate of $\mathcal{Q}(\mathbf{w}^\star)$.

The *inverse multi-armed bandit problem* is to recover $\mathbf{w}^\star$ given the distribution $\tilde{\mathcal{Q}}$, which allows us to identify the optimal arm. In order to achieve this, we aim to find $\mathbf{w}$ such that $\mathcal{Q}(\mathbf{w}) = \tilde{\mathcal{Q}}$, or matches it as closely as possible. Ideally, we would want to find $\mathbf{w}$ such that $\mathcal{Q}(\mathbf{w}) = \mathcal{Q}(\mathbf{w}^\star)$,[3] but since we do not have access to $\mathcal{Q}(\mathbf{w}^\star)$, we use the unbiased estimate $\tilde{\mathcal{Q}}$ in its place.[4] Below, we produce conditions under which the optimal policy is recoverable, and provide a practical algorithm that achieves this for all settings that meet the criteria.

Identifying the Optimal Arm

Since the constraint $\mathcal{Q}(\mathbf{w}) = \tilde{\mathcal{Q}}$ is "far" from being convex in $\mathbf{w}$, we reformulate the problem such that the new problem is convex, and all its optimal solutions satisfy the required constraint (and vice versa). The new objective we use is the cross entropy loss between $\tilde{\mathcal{Q}}$ and $\mathcal{Q}(\mathbf{w})$. That is, the optimization problem to solve is

$$\min_{\mathbf{w}} - \sum_{k \in A} \tilde{\mathcal{Q}}_k \log \mathcal{Q}(\mathbf{w})_k. \tag{1}$$

[3] Note that there might be multiple $\mathbf{w}$ such that $\mathcal{Q}(\mathbf{w}) = \mathcal{Q}(\mathbf{w}^\star)$. However, since we care only about the corresponding optimal arm, and identifiability tells us that all weight vectors with the same $\mathcal{Q}$ value have the same optimal arm, we just need to find one such weight vector.

[4] In most cases we will have collected sufficient data such that the optimal arm corresponding to $\tilde{\mathcal{Q}}$ coincides with the optimal arm corresponding to $\mathcal{Q}(\mathbf{w}^\star)$. Although they may not coincide, this probability goes to zero as the size of the dataset $\mathcal{D}$ increases.

It is obvious that this objective is optimized at points with $\mathcal{Q}(\mathbf{w}) = \tilde{\mathcal{Q}}$, if the original problem was feasible. Otherwise, it finds $\mathbf{w}$ whose $\mathcal{Q}$ is as close to $\tilde{\mathcal{Q}}$ as possible in terms of cross-entropy. Furthermore, this optimization problem is convex under a simple condition, which requires the definition of X_k as an $(m-1) \times d$ matrix with rows of the form $(\mathbf{x}_k - \mathbf{x}_j)^\intercal$, for each $j \in A \setminus \{k\}$.

Theorem 0.6. *Optimization problem* (1) *is convex if $X_k X_k^\intercal$ is invertible for each $k \in A$.*

The proof of the theorem appears in the full version of the paper. An exact characterization of when $X_k X_k^\intercal$ is full rank is $\text{rank}(X_k X_k^\intercal) = \text{rank}(X_k) = m - 1$, i.e. when X_k is full row rank. For this to be true, a necessary condition is that $d \geq m - 1$ as $\text{rank}(X_k) \leq \min(d, m-1)$. And under this condition, the requirement for X_k to to be full row rank is that the rows $(\mathbf{x}_k - \mathbf{x}_j)^\intercal$ are linearly independent, which is very likely to be the case, unless the feature vectors were set up adversarially. One potential scenario where the condition $d \geq m - 1$ would arise is when there are many features but feature vectors $\mathbf{x}_j$ are sparse.

As the optimization problem (1) is convex, we can use gradient descent to find a minimizer. And for this, we need to be able to compute the gradient accurately, which we show is possible; the calculation is given in the full version of the paper.

Importantly, we can also use our procedure to determine whether the optimal arm is identifiable. Given $\tilde{\mathcal{Q}}$, we solve the optimization problem (1) to first find a $\mathbf{w}_o$ such that $\mathcal{Q}(\mathbf{w}_o) = \tilde{\mathcal{Q}}$. Let $\mathbf{w}_o$ have the optimal arm $a_o \in A$. Now, our goal is to check if there exists any other weight $\mathbf{w}$ that has $\mathcal{Q}(\mathbf{w}) = \tilde{\mathcal{Q}}$ but whose corresponding optimal arm is not a_o. To do this, we can build a set of convex programs, each with the exact same criterion (taking care of the $\mathcal{Q}(\mathbf{w}) = \tilde{\mathcal{Q}}$ requirement), but with the constraint that arm $a_i \neq a_o$ is the optimal arm (or at least beats a_o) with respect to $\mathbf{w}$. In particular, the constraint for program i could be $\mathbf{w}^\intercal \mathbf{x}_i > \mathbf{w}^\intercal \mathbf{x}_{a_o}$.[5] As this is a simple affine constraint, solving the convex program is very similar to running gradient descent as before. If any of these convex programs outputs an optimal solution that satisfies $\mathcal{Q}(\mathbf{w}) = \tilde{\mathcal{Q}}$, then the problem is not identifiable, as it implies that there exist weight vectors with different optimal arms leading to the same $\tilde{\mathcal{Q}}$. On the other hand, if none of them satisfies $\mathcal{Q}(\mathbf{w}) = \tilde{\mathcal{Q}}$, we can conclude that a_o is the desired unique optimal arm.

Experiments

We next study the empirical performance of our algorithm for the inverse multi-armed bandit problem. We focus on instances inspired by the counter-example from Lemma 0.4. The reason for this is that in randomly generated bandit problems, the optimal arm $a^\star$ is very likely to be the mode of $\mathcal{Q}(\mathbf{w}^\star)$, making the mode of $\tilde{\mathcal{Q}}$ a very good estimator of $a^\star$.[6] By contrast, the counterexample allows us to generate "hard" instances.

Specifically, the bandit instances we consider have two features ($d = 2$) and three arms $A = \{1, 2, 3\}$, and their features are defined as $\mathbf{x}_1 = [1, 1]$, $\mathbf{x}_2 = [2, -\delta]$ and $\mathbf{x}_3 = [-\delta, 2]$, where $\delta > 0$ is a positive constant. The ground truth weight vector is given as $\mathbf{w}^\star = [1, 1]$. Hence, for any $\delta > 0$, the optimal arm is arm 1. The noise is $\boldsymbol{\eta} \sim \mathcal{N}(0, \sigma^2)$. Such an instance is very similar to the one of Lemma 0.4, except that the features are not replicated to extend from two to four features, and hence the problem remains identifiable.

Observe that when the value of δ is small enough, the blue region of Figure 1 becomes a sliver, capturing a very small density of the noise $\boldsymbol{\eta}$, and causing arm 1 to not be the mode of $\mathcal{Q}(\mathbf{w}^\star)$. Alternatively, for a given value of δ, if σ is large enough, most of the noise's density escapes the blue region, again causing arm 1 to not be the mode of $\mathcal{Q}(\mathbf{w}^\star)$. In the following experiments, we vary both δ and σ, and show that even when the optimal arm almost never appears in $\mathcal{Q}(\mathbf{w}^\star)$, our algorithm is able to recover it.

Varying parameter δ. In the first set of experiments, we fix the noise standard deviation σ to 1, generate $n = 500$ agents according to the noise $\boldsymbol{\eta} \sim \mathcal{N}(0, \sigma^2)$, and vary parameter δ from 0.01 to 3. Figure 2 shows the percentage of times our algorithm and the mode recover the optimal arm 1. This graph is averaged over 1000 runs, and error bars depict 95% confidence intervals.

When δ is extremely close to 0, the optimal arm's region almost vanishes. Hence, small differences between $\tilde{\mathcal{Q}}$ and $\mathcal{Q}(\mathbf{w}^\star)$ could have a substantial effect, and unless $\mathbf{w}^\star$ is numerically recovered within this sliver, the optimal arm would not be recovered. As we move to even slightly larger values of δ, however, the performance of the algorithm improves substantially and it ends up recovering the optimal arm 100% of the time.

By contrast, as δ is varied from 0 to ∞, the density of the noise $\boldsymbol{\eta}$ captured by the blue region increases continuously from 0 to that of the first quadrant. In particular, there is a point where $\mathcal{Q}(\mathbf{w}^\star)$ has probability tied across the three arms, after which arm 1 is always the mode (i.e. mode has 100% performance), and before which arms 2 and 3 are the modes (i.e the mode has 0% performance). This tipping point is evident from the graph and occurs around $\delta = 1$.[7] Observe that the performance of the algorithm rises to 100% much before this tipping point, serving as evidence that it can perform well even if the optimal arm barely appears in the dataset. Similar results on when the parameters are set to $\sigma \in \{0.5, 2.0\}$ or $n \in \{250, 1000\}$ may be found in the full

[5] The strong inequality can be implemented in the standard way via $\mathbf{w}^\intercal \mathbf{x}_i \geq \mathbf{w}^\intercal \mathbf{x}_{a_o} + \epsilon$ for a sufficiently small $\epsilon > 0$ that depends on the program's bit precision.

[6] This is because, for each arm a, the region $\mathcal{R}_a = \{\mathbf{w} : \mathbf{w}^\intercal \mathbf{x}_a \geq \mathbf{w}^\intercal \mathbf{x}_j \text{ for each } j\}$, corresponding to where arm a is optimal, forms a polytope, and the optimal arm's region $\mathcal{R}_{a^\star}$ contains $\mathbf{w}^\star$. Hence, as long as $\mathcal{R}_{a^\star}$ has enough volume around $\mathbf{w}^\star$, it would capture a majority of the density of the noise $\boldsymbol{\eta}$, and $a^\star$ would be the mode of the distribution $\mathcal{Q}(\mathbf{w}^\star)$.

[7] The transition in this graph is smoother than a step function because we use the empirical mode from $\tilde{\mathcal{Q}}$ whose performance varies smoothly as the distance between probabilities of arms 1 and $\{2, 3\}$ changes.

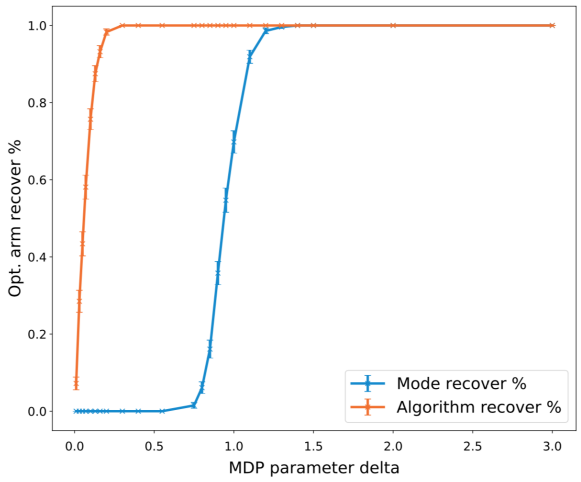


Figure 2: Performance as δ is varied.

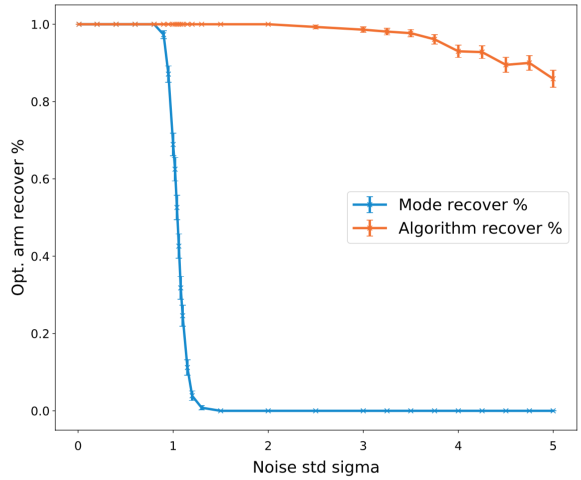


Figure 3: Performance as σ is varied.

version of the paper.

Varying noise parameter σ. Next, we fix the parameter δ to 1 and generate $n = 500$ agents according to noise $\boldsymbol{\eta} \sim \mathcal{N}(0, \sigma^2)$, while varying the noise parameter σ from 0.01 to 5. Figure 3 shows the percentage of times our algorithm and the mode recover the optimal arm 1. This graph is also averaged over 1000 runs, and error bars depict 95% confidence intervals.

The results are similar in spirit to Figure 2. When σ is extremely large (relative to the ground truth vector $\mathbf{w}^\star = [1, 1]$), the weight space becomes less and less distinguishable with respect to the corresponding $\mathcal{Q}$ values. In particular, small differences between $\tilde{\mathcal{Q}}$ and $\mathcal{Q}(\mathbf{w}^\star)$ again have a substantial effect on the corresponding optimal arms, causing a suboptimal arm to be recovered. At more reasonable levels of noise, however, we can see that the algorithm recovers the optimal arm 100% of the time.

The mode's performance also has a similar flavor to Figure 2. For a given value of δ, the regions of Figure 1 are completely decided. When σ is close to zero, the noise is almost negligible, and hence the blue region captures most of the density of the noise $\boldsymbol{\eta}$, and the optimal arm is the mode. But as σ is varied from 0 to ∞, the density captured by this region decreases continuously from 1 to a ratio of the volumes of the regions. In particular, we again come across a point where $\mathcal{Q}(\mathbf{w}^\star)$ has probability tied across the three arms, before which arm 1 is always the mode (i.e. mode has 100% performance), and after which arms 2 and 3 are the modes (i.e. the mode has 0% performance). Note that, for $\sigma = 1$, this point is achieved around $\delta = 1$ (Figure 2). Hence, when we vary σ while fixing $\delta = 1$, the tipping point is expected to be achieved around $\sigma = 1$, which is indeed the case, as evident from Figure 3. Again, observe that the performance of the algorithm is still around 100% significantly after this tipping point. Similar results on when the parameters are set to $\delta \in \{0.5, 2.0\}$ or $n \in \{250, 1000\}$ may be found in the full version of the paper.

Discussion

We have shown that it is possible to match the performance of the uniform mixture π^u, or that of the average agent. In Section we then established that it is possible to learn policies from demonstrations with *superior* performance compared to the teacher, albeit under simplifying assumptions. An obvious challenge is to relax the assumptions, but this is very difficult, and we do not know of existing work that can be applied directly to our general setting. Indeed, the most relevant theoretical work is that of Syed and Schapire (2008). Their approach can only be applied if the sign of the reward weight is known for every feature. This is problematic in our setting as some agents may consider a feature to be positive, while others consider it to be negative. A priori, it is unclear how the sign can be determined, which crucially invalidates the algorithm's theoretical guarantees. Moreover, it is unclear under which cases the algorithm would produce a policy with superior performance, or if such cases exist.

We also remark that, although in the general setting we seek to compete with π^u, we are actually doing something quite different. Indeed, *ex post* (after the randomness has been instantiated) the uniform mixture π^u simply coincides with one of the individual policies. By contrast, IRL algorithms pool the feature expectations of the trajectories $\tau_1, \ldots, \tau_n$ together, and try to recover a policy that approximately matches them. Therefore, we believe that IRL algorithms do a much better job of aggregating the individual policies than π^u does, while giving almost the same optimality guarantees.

Finally, we wish to highlight potential extensions to our work. One promising extension is to better understand how notions of social choice could be folded into our framework of aggregation through IRL; we include a short discussion of this point in the full version of the paper. Another promising direction is to understand what can be gained (e.g., in terms of sample complexity) by going beyond the classical setup of IRL, for instance by allowing for interaction and/or communication between the agents and teachers.

Acknowledgments

This work was partially supported by the National Science Foundation under grants CCF-2007080, IIS-2024287 and CCF-1733556; and by the Office of Naval Research under grant N00014-20-1-2488. Tom Yan is supported by a U.S. National Science Foundation Graduate Research Fellowship.

Ethics Statement

As mentioned in Section , our work can conceivably be used to align the values of an AI system with those of a group of people or even those of society. Although such an application would be far in the future, we acknowledge that it gives rise to ethical issues. Most notably, in cases where the values of the set of agents are not centered around a reasonable moral system, learning from those agents may lead to undesirable behavior. Thought must be given both to the choice of appropriate applications as well as to the selection of agents who serve as teachers.

References

Abbeel, P.; and Ng, A. Y. 2004. Apprenticeship Learning via Inverse Reinforcement Learning. In *Proceedings of the 21st International Conference on Machine Learning (ICML)*, 1–8.

Babeş-Vroman, M.; Marivate, V.; Subramanian, K.; and Littman, M. L. 2011. Apprenticeship Learning About Multiple Intentions. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, 897–904.

Choi, J.; and Kim, K.-E. 2012. Nonparametric Bayesian Inverse Reinforcement Learning for Multiple Reward Functions. In *Proceedings of the 25th Annual Conference on Neural Information Processing Systems (NeurIPS)*, 314–322.

Chomsky, N. 1965. *Aspects of the Theory of Syntax*. MIT Press.

Everitt, T.; Krakovna, V.; Orseau, L.; and Legg, S. 2017. Reinforcement Learning with a Corrupted Reward Channel. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*, 4705–4713.

Givan, R.; Leach, S.; and Dean, T. 2000. Bounded-Parameter Markov Decision Processes. *Artificial Intelligence* 122(1–2): 71–109.

Hadfield-Menell, D.; Russell, S. J.; Abbeel, P.; and Dragan, A. D. 2016. Cooperative Inverse Reinforcement Learning. In *Proceedings of the 30th Annual Conference on Neural Information Processing Systems (NeurIPS)*, 3909–3917.

Hodges Jr, J. L.; and Lehmann, E. L. 1950. Some problems in minimax point estimation. *The Annals of Mathematical Statistics* 182–197.

Kaelbling, L. P.; Littman, M. L.; and Moore, A. W. 1996. Reinforcement Learning: A Survey. *Journal of Artificial Intelligence Research* 4: 237–285.

Kober, J.; Bagnell, J. A.; and Peters, J. 2013. Reinforcement Learning in Robotics: A Survey. *International Journal of Robotics Research* 32(11): 1238–1274.

Mikhail, J. 2011. *Elements of Moral Cognition: Rawls' Linguistic Analogy and the Cognitive Science of Moral and Legal Judgment*. Cambridge University Press.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; Petersen, S.; Beattie, C.; Sadik, A.; Antonoglou, I.; King, H.; Kumaran, D.; Wierstra, D.; Legg, S.; and Hassabis, D. 2015. Human-Level Control Through Deep Reinforcement Learning. *Nature* 518: 529–533.

Ng, A. Y.; and Russell, S. 2000. Algorithms for Inverse Reinforcement Learning. In *Proceedings of the 17th International Conference on Machine Learning (ICML)*, 663–670.

Noothigattu, R.; Gaikwad, S. S.; Awad, E.; Dsouza, S.; Rahwan, I.; Ravikumar, P.; and Procaccia, A. D. 2018. A Voting-Based System for Ethical Decision Making. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI)*, 1587–1594.

Perron, F.; and Marchand, E. 2002. On the minimax estimator of a bounded normal mean. *Statistics and Probability Letters* 58: 327–333.

Rawls, J. 1971. *A Theory of Justice*. Harvard University Press.

Regan, K.; and Boutilier, C. 2009. Regret-based Reward Elicitation for Markov Decision Processes. In *Proceedings of the 25th Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, 444–451.

Regan, K.; and Boutilier, C. 2010. Robust Policy Computation in Reward-uncertain MDPs using Nondominated Policies. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI)*, 1127–1133.

Russell, S.; Dewey, D.; and Tegmark, M. 2015. Research Priorities for Robust and Beneficial Artificial Intelligence. *AI Magazine* 36(4): 105–114.

Silver, D.; Huang, A.; Maddison, C. J.; Guez, A.; Sifre, L.; van den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; Dieleman, S.; Grewe, D.; Nham, J.; Kalchbrenner, N.; Sutskever, I.; Lillicrap, T.; Leach, M.; Kavukcuoglu, K.; Graepel, T.; and Hassabis, D. 2016. Mastering the Game of Go with Deep Neural Networks and Tree Search. *Nature* 529: 484–489.

Sutton, R. S.; and Barto, A. G. 1998. *Reinforcement Learning: An Introduction*. MIT Press.

Syed, U.; and Schapire, R. E. 2008. A Game-Theoretic Approach to Apprenticeship Learning. In *Proceedings of the 21st Annual Conference on Neural Information Processing Systems (NeurIPS)*, 1449–1456.

Zheng, J.; Liu, S.; and Ni, L. M. 2014. Robust Bayesian Inverse Reinforcement Learning with Sparse Behavior Noise. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI)*, 2198–2205.

Ziebart, B. D.; Maas, A. L.; Bagnell, J. A.; and Dey, A. K. 2008. Maximum Entropy Inverse Reinforcement Learning. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence (AAAI)*, 1433–1438.