If MaxEnt RL is the Answer, What is the Question?

Benjamin Eysenbach [1 2] **Sergey Levine** [1 3]

Abstract

Experimentally, it has been observed that humans and animals often make decisions that do not maximize their expected utility, but rather choose outcomes randomly, with probability proportional to expected utility. Probability matching, as this strategy is called, is equivalent to maximum entropy reinforcement learning (MaxEnt RL). However, MaxEnt RL does not optimize expected utility. In this paper, we formally show that MaxEnt RL does optimally solve certain classes of control problems with variability in the dynamics and reward function. In particular, we show (1) that MaxEnt RL is equivalent to a two-player game where an adversary chooses the dynamics and reward function, and (2) that MaxEnt RL can be used to solve a certain class of POMDPs. These results suggest a deeper connection between MaxEnt RL, robust control, and POMDPs.

1. Introduction

Reinforcement learning (RL) searches for a policy that maximizes the expected, cumulative reward. In fully observed Markov decision processes (MDPs), this maximization always has a deterministic policy as a solution. Maximum entropy (MaxEnt) RL is a modification of the RL objective that further adds an entropy term to the objective. This additional entropy term causes MaxEnt RL to seek policies that are stochastic and have a non-zero probability of sampling every action. MaxEnt RL has appealing connections to probabilistic inference (Dayan and Hinton, 1997; Neumann et al., 2011; Todorov, 2007; Kappen, 2005; Toussaint, 2009; Rawlik et al., 2013; Theodorou et al., 2010; Ziebart, 2010), prompting a renewed interest in recent years (Haarnoja et al., 2018b; Abdolmaleki et al., 2018; Levine, 2018). MaxEnt RL can also be viewed as using Thompson sampling (Thompson, 1933) to collect trajectories, where the posterior belief is given by the exponentiated return. Empirically, MaxEnt RL algorithms achieve good performance on a number of simulated (Haarnoja et al., 2018b) and real-world (Haarnoja et al., 2018a; Singh et al., 2019) control tasks, and can be more robust to perturbations (Haarnoja et al., 2018c).

The empirical success of MaxEnt RL algorithms on RL problems is surprising, as MaxEnt RL optimizes a different objective than standard RL. The solution to every MaxEnt RL problem is stochastic, while deterministic policies can always be used to solve standard RL problems (Puterman, 2014). It remains an open question as to whether the standard MaxEnt RL objective actually optimizes some well-defined notion of risk or regret that would account for its observed empirical benefits. This paper studies this problem, and aims to answer the following question: *if MaxEnt RL is the solution, then what is the problem?*

In this paper, we show that MaxEnt RL provides the optimal policy in settings with uncertainty and variability in the reward function. More precisely, we show that MaxEnt RL is equivalent to two separate, challenging problems: (1) *robust control* and (2) regret minimization in a *meta-POMDP*. In the first setting, robust control, we consider an adversary that chooses some aspects of the dynamics or reward function. Intuitively, we expect stochastic policies to be most robust because they are harder to exploit, as we formalize in Sec. 4. The second setting, the meta-POMDP, is a partially observed MDP where the reward depends on an unobserved portion of the state, and where multiple episodes in the original MDP correspond to a single extended trial in the meta-POMDP. This problem setting reflects real-world settings where the aim is to solve a task in as few trials as possible. While both robust control and regret minimization in a meta-POMDP are natural problems that arise in many real-world scenarios, neither is an expected utility maximization problem, so we cannot expect optimal control to solve these problems. In contrast, we show that MaxEnt RL provides solutions to both problems.

2. Preliminaries

We begin by defining notation and discussing some previous motivations for MaxEnt RL. An agent observes states s_t, takes actions $a_t \sim \pi(a_t \mid s_t)$, and obtains rewards $r(s_t, a_t)$. The initial state is sampled $s_1 \sim p_1(s_1)$, and subsequent states are sampled $s' \sim p(s' \mid s, a)$. Episodes

*Equal contribution [1]Google Brain [2]Carnegie Mellon University [3]UC Berkeley. Correspondence to: Benjamin Eysenbach <beysenba@cs.cmu.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Vienna, Austria, PMLR 108, 2020.

have T steps, which we summarize as a trajectory $\tau \triangleq (s_1, a_1, \cdots, s_T, a_T)$. The MaxEnt RL problem, also known as the entropy-regularized control problem, is to maximize the sum of expected reward and conditional action entropy, $\mathcal{H}_\pi[a \mid s]$:

$$J_{\text{MaxEnt}}(\pi; p, r) \triangleq \mathbb{E}_{\substack{a\sim\pi(a|s) \\ s'\sim p(s'|s,a)}}\left[\sum_{t=1}^{T} r(s_t, a_t) + \mathcal{H}_\pi[a_t \mid s_t]\right], \tag{1}$$

where the objective is parameterized by the dynamics p and reward function r. Prior work on MaxEnt RL offers a slew of intuitive explanations for why one might prefer MaxEnt RL. We summarize three common explanations and highlight problems with each in Appendix A

3. What Problems Does MaxEnt RL Solve?

In general, all fully observed MDPs admit deterministic optimal policies for the expected reward objective (Puterman, 2014), so we might wonder why stochastic policies would ever be preferred. The central theme of this paper is that stochastic policies are preferable in scenarios that involve *regret minimization*. Such scenarios arise in robust control problems, where the agent is must cope with adversarial disturbances, and in problems where the agent must repeatedly attempt a task under partial observability until it succeeds. In both settings, a policy that never plays a potentially necessary strategy could incur infinite regret: in the adversarial case, the adversary will always select the task for which the policy is weakest; in the partially observed setting, if the policy never attempts a certain strategy, the policy will get stuck when trying to solve tasks that can only be solved by that certain strategy (and thus accumulate infinite regret). We will formalize these two settings below, and the subsequent two sections will describe how MaxEnt RL can learn optimal policies for these two settings.

Answer 1: Robustness to Adversarial Perturbations. The first strength of stochastic policies is that they are robust to adversarial perturbations. For example, in the game rock-paper-scissors ("ro-sham-bo"), it is bad to always choose the same action (say, rock) because an adversary can always choose an action that makes the player perform poorly (e.g., by choosing paper). Robustness is useful even in the absence of adversaries, as it ensures that policies perform well in settings with slightly different rewards and dynamics.

Answer 2: Partially Observed MDPs. The second strength of stochastic policies is that they avoid waiting infinitely long to find a good outcome. Imagine that a cookie is hidden in one of two jars. A policy that always looks in jar A will never find a cookie hidden in jar B; such a policy would incur infinite regret. This need to try various approaches arises in many realistic settings where we do not get to observe the true reward function, but rather have a belief over what the true reward is. Given this belief, we want to determine the true reward as quickly as possible. For example, in a health-care setting, a physician may want to find a course of treatment to cure a patient. Which treatment will cure the patient depends on their illness, which is unknown. A physician will prescribe medications based on his beliefs about the patient's illness. If the medication fails, the patient returns to the physician the next week and the physician recommends another medication, a process that continues until the patient is cured. Note that a physician that always prescribed the same medication would fail to cure many patients. In Sec. 5, we define a meta-level POMDP for describing these sorts of tasks and show that MaxEnt RL *minimizes regret* in such settings.

4. MaxEnt RL and Robust Control

In this section, we will formalize the intuition that MaxEnt RL yields robust policies by proving that MaxEnt RL effectively bounds regret against adversarial perturbations to the dynamics and rewards. Proofs are given in Appendix C.

4.1. Robustness to Adversarial Dynamics

We start by showing that MaxEnt RL is robust against adversarial disturbances to the *dynamics*. We will be interested in the performance of policy π under dynamics $\tilde{p}$ chosen by the adversary.

Theorem 4.1. *Let an MDP with dynamics $p(s' \mid s, a)$ and reward function $r(s, a) > 0$ be given. Define the transformed reward function $\bar{r}(s, a) \triangleq \log r(s, a) + \mathcal{H}[s' \mid s, a]$. Then there exists a constant $\epsilon > 0$ such that the MaxEnt RL objective J_{MaxEnt} (Eq. 1) with dynamics p and reward function $\bar{r}$ maximizes a lower bound on the robust control objective defined by robust set $\tilde{P}$:*

$$\min_{\tilde{p}\in\tilde{\mathcal{P}}(\pi)} J_{\text{MaxEnt}}(\pi; \tilde{p}, r) \geq \exp(J_{\text{MaxEnt}}(\pi; p, \bar{r})),$$

where the set of dynamics that the adversary chooses from, $\tilde{\mathcal{P}}$, is defined as

$$\left\{\tilde{p} \;\middle|\; \mathbb{E}_{\substack{a\sim\pi(a|s), \\ s'\sim p(s'|s,a)}}\left[\sum_t \log \iint e^{\log p(s'|s,a) - \log \tilde{p}(s'|s,a)} da'_t ds'_{t+1}\right] \leq \epsilon\right\}.$$

This result implies that, if we wish to optimize reward function r and be robust against perturbations to the dynamics, we can run MaxEnt RL with the reward function $\bar{r}(s, a) = \log r(s, a) + \mathcal{H}[s' \mid s, a]$. In many settings (e.g., dynamics with constant additive noise), we might assume that $\mathcal{H}[s' \mid s, a]$ is constant, in which case we simply set the MaxEnt RL reward to $\bar{r}(s, a) = \log r(s, a)$.

Example. Consider a MDP with 1D, bounded, states and actions $s, a \in [-10, 10]$. The dynamics are $p(s' \mid s, a) = \mathcal{N}(s'; \mu = As + Ba, \sigma = 1)$ and the reward function is $r(s, a) = \|s\|_2^2$. Note that the dynamics entropy $\mathcal{H}[s' \mid s, a]$ is constant, so we can ignore it when constructing the MaxEnt RL reward function: $\bar{r}(s, a) = \frac{1}{2} \log \|s\|_2^2$.

We will consider an adversary that modifies the dynamics by increasing the standard deviation to $\sqrt{2}$ and shifts the bias by an amount β, resulting in the following dynamics: $\tilde{p}(s' \mid s, a) = \mathcal{N}(s'; \mu = As + Ba + \beta, \sigma = \sqrt{2})$. The penalty for the adversary is $\frac{1}{2}\beta^2 + \log(8\sqrt{\pi}) + \log(20)$, so the robust set corresponds to dynamics where the mean shifts by at most $\mathcal{O}(\sqrt{\epsilon})$. Theorem 4.1 says that we can obtain a policy that is robust against this set of dynamics by simply applying MaxEnt RL to the reward function $\bar{r}$.

4.2. Robustness to Adversarial Reward Functions

In this section, we will show that MaxEnt RL is robust against adversarial perturbations to the reward function. We use $\tilde{r}$ to denote the adversarial reward function.

Theorem 4.2. *For any dynamics p and reward function r, there exists a positive constant $\epsilon > 0$ such that the MaxEnt RL objective J_{MaxEnt} is equivalent to the robust control objective defined by robust set $\tilde{\mathcal{R}}(\pi)$:*

$$\min_{\tilde{r} \in \tilde{\mathcal{R}}(\pi)} \mathbb{E}\Big[\sum_t \tilde{r}(s_t, a_t)\Big] = J_{MaxEnt}(\pi; p, r) \quad \forall \pi,$$

where

$$\tilde{\mathcal{R}}(\pi) \triangleq \left\{ \tilde{r} \;\middle|\; \mathbb{E}\Big[\sum_t \log \int \exp(r(s_t, a_t) - \tilde{r}(s_t, a_t)) da'_t\Big] \le \epsilon \right\}. \tag{2}$$

This results says that the policy obtained by applying MaxEnt RL to reward function r is robust against adversarial disturbances to this reward function.

Example 1. We use a task with a 1D, bounded action space $\mathcal{A} = [-10, 10]$ and a reward function composed of a task-specific reward r_{task} and a penalty from deviating from some desired action a^*:

$$r(s, a) \triangleq r_{\text{task}}(s, a) - (a - a^*)^2.$$

The adversary will perturb this desired action by an amount Δa and decrease the weight on the control penalty by 50%, resulting in the following reward function:

$$\tilde{r}(s, a) \triangleq r_{\text{task}}(s, a) - \frac{1}{2}(a - (a^* + \Delta a))^2.$$

In this example, the penalty is $\Delta a^2 + \frac{1}{2}\log(2\pi) + \log(20)$. See Appendix C.4 for the full derivation. Thus, up to an additive constant, the penalty corresponds to the squared amount by which the prior action is shifted. In this example, Theorem 4.2 says that MaxEnt RL with reward function r yields a policy that is robust against adversaries that shift a^* by at most $\mathcal{O}(\sqrt{\epsilon})$.

Example 2: continuous control locomotion We next apply MaxEnt RL on four continuous control tasks. We use four continuous control tasks from OpenAI Gym (Brockman et al., 2016). As shown in Fig. 1, only MaxEnt RL succeeds as maximizing the worst-case reward.

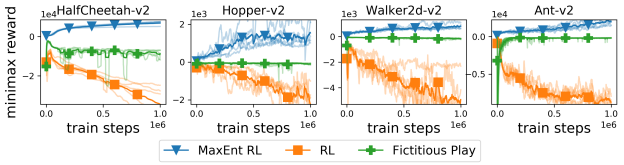

Figure 1. **MaxEnt RL solves robust-reward control problems for robotic control tasks.** Only MaxEnt RL succeeds at maximizing worst-case reward.

5. MaxEnt RL and POMDPs

In this section, we describe how solving an unknown task can be viewed as POMDP. Regret minimization in partially observed settings with unknown tasks is closely related to regret minimization in the face of an adversary: if the tasks are selected adversarially, then the policy will incur infinite regret if the adversary selects a task that it cannot perform. This section builds on this idea to show that MaxEnt RL provides the optimal solution for these POMDPs.

5.1. Defining the Meta-POMDP

A *meta-POMDP* is a POMDP defined in terms of a MDP together with a distribution over tasks (i.e., rewards). Each step in the meta-POMDP corresponds to an entire episode in the MDP. The agent's observation is an entire trajectory τ from the MDP. The state in the meta-POMDP is a combination of the observed trajectory τ and an *unobserved* task τ^*. We consider a fully general definition of tasks as matching a desired trajectory τ^*, noting that this definition subsumes many other tasks families. The agent's actions will be deterministic policies π. Note that stochastic policies, like those learned by MaxEnt RL, implicitly define a distribution over deterministic policies (Ng and Jordan, 2013). Episodes in the meta-POMDP start by sampling a task $\tau^* \sim p(\tau^*)$ from the task distribution $p(\tau^*)$, and end when the agent has matched the desired trajectory. Section 5.2 will discuss how these task distributions can be constructed from reward functions, and vice versa. The agent's reward function is +1 if it matches the desired trajectory and 0 otherwise. To avoid confusion, we will use "meta-step" and "meta-episode" to refer to steps and episodes in the meta-POMDP.

We define the cumulative *regret* of an agent as the expected number of meta-steps required to complete the task. For example, in the health-care example, the regret is the number of times the patient visits the physician before being cured. Formally, the number of episodes for policy π to solve task τ^* is a geometric random variable with expected value is $1/\pi(\tau^*)$, so the policy's regret is:

$$\text{Regret}_p(\pi) = \mathbb{E}_{\tau^* \sim p}\left[1/\pi(\tau^*)\right]. \tag{3}$$

Discussion. The meta-POMDP is not a standard RL problem: the optimal policy for the meta-POMDP continues to explore throughout the meta-episode, trying new approaches

Table 1. **Examples of meta-POMDPs**

	Task Distribution	MaxEnt RL Reward
Bandit	$p(a^*)$	$\frac{1}{2}\log p(a)$
Goal reaching	$p(s_T^*)$	$\frac{1}{2}\mathbb{1}(t=T)\log p(s_t)$
Markovian	$\prod_t p_\tau(s_t, a_t)$	$\frac{1}{2}\log p(s_t, a_t)$
Non-Markovian	$p(\tau)$	$\frac{1}{2}\log p(\tau)$

until the task is solved. In contrast, a standard RL agent would use the same approach for every meta-step, incurring infinite regret if this approach fails. Our analysis of the meta-POMDP will consider Markovian strategies for solving the meta-POMDP: the agent keeps sampling trajectories until it stumbles upon the target trajectory. *The agent does not perform any adaptation.*

5.2. Solving the Meta-POMDP

This section draws a bridge between MaxEnt RL and the meta-POMDP. The Markovian assumption above means that solutions to a meta-POMDP can be represented as a stochastic policy (i.e., a distribution over deterministic policies (Ng and Jordan, 2013)). In this section we will show that the stochastic policy produced by MaxEnt RL is itself the solution to a meta-POMDP. Table 1 summarizes four types of increasingly-general meta-POMDPs. We discuss the first three types below.

Bandit meta-POMDPs. First, we examine meta-POMDPs where the underlying MDP is a bandit problem. In these bandit meta-POMDPs, tasks correspond to sampling a target action $a^* \sim p(a^*)$. Intuitively, bandit meta-POMDPs are multi-armed bandits with one (unknown) arm which yields a payout; the agent's aim is to get the payout with as few arm pulls as possible. The policy that minimizes regret (Eq. 3) is $\pi(a) \propto \sqrt{p(a)}$. For the cookie jar example in Sec. 3, this result says that the optimal strategy is to randomly choose which jar to open, but to prefer whichever jar is more likely to contain the cookie. We can learn this optimal policy using the reward function $r(s, a) = \frac{1}{2}\log p(a)$.

Goal-reaching meta-POMDPs. Moving beyond bandits, we now examine meta-POMDPs without any restrictions on the underlying MDP and where tasks correspond to reaching a particular (unknown) goal state. While prior work on goal reaching (Kaelbling, 1993; Schaul et al., 2015; Andrychowicz et al., 2017) typically assumes that the goal state is observed, here will assume that the goal state $s^* \sim p(s^*)$ is not observed. For simplicity, we will only consider the task solved if the agent remains at the goal at the final time step. The agent's aim is to reach the goal in as few episodes as possible. For example, imagine an ice cream scooping robot serving a customer. The customer has a desired "goal" flavor, which they are unable to express to the robot (perhaps they are too young to speak; perhaps they have simply forgotten the name of the flavor). Based on past experience, the robot has a prior belief over which flavors the customer will prefer. The robot hands the customer samples of different flavors until finding the desired flavor.

To solve goal-reaching meta-POMDPs, we can simply apply MaxEnt RL with the reward function $r(s_t, a_t) = \frac{1}{2}\mathbb{1}(t = T) \cdot \log p(s_t)$. Again, we can read this statement in reverse: applying MaxEnt RL to tasks with rewards provided at only the terminal state yield policies that are *optimal* for reaching unknown goals in as few trials as possible, where the goal distribution is defined as $p(s^*) = \exp(2 \cdot r(s^*))$.

Markovian meta-POMDPs. Finally, we look at the even more general Markovian meta-POMDP, where the task distribution can be an arbitrary function of transitions: $p(\tau) = \prod_{t=1}^{T-1} p_\tau(s_t, a_t)$. These factors encode a prior belief over which states and actions are likely to be included or excluded from successful trajectories. For example, imagine a robot navigating across a field of landmines. Our aim is *not* for the robot to minimize the number of explosions in a single episode, but rather to minimize the number of *episodes* needed to cross the field without running over a landmine.

Solving Markovian meta-POMDPs with MaxEnt RL is straightforward: just apply MaxEnt RL to the reward function $r(s_t, a_t) = \frac{1}{2}\log p_\tau(s_t, a_t)$. Any (Markovian) task can be converted into an equivalent meta-POMDP, so we can interpret arbitrary MaxEnt RL problems as solving a certain (Markovian) meta-POMDP. Given an arbitrary reward function $r(s, a)$ and making the same assumptions as before, MaxEnt RL with this reward function is implicitly solving the factored meta-POMDP defined by $p_\tau(s_t, a_t) = \exp(2 \cdot r(s_t, a_t))$. The equivalence between MaxEnt RL and Markovian meta-POMDPs suggests that the policies produced by MaxEnt RL will be able to solve new tasks with a small number of trials. Appendix D.4 discusses non-Markovian meta-POMDPs, and Appendix D.6 provides a computational experiment.

6. Discussion

While stochastic policies are not optimal for maximizing reward, they can be optimal at minimizing specific types of regret, as in the robust control and meta-POMDP settings. These two settings have close connections: in both cases, stochastic policies are optimal when the goal is to minimize or bound regret, and an *exploitable* policy (i.e., one that never visits some potential goal or performs poorly on one of the rewards in the robust set) can incur infinite regret. We hope that this analysis sheds light on the potential utility of stochastic MaxEnt RL policies, particularly to real-world applications, where robustness can be especially important.

References

Abdolmaleki, A., Springenberg, J. T., Tassa, Y., Munos, R., Heess, N., and Riedmiller, M. (2018). Maximum a posteriori policy optimisation. *arXiv preprint arXiv:1806.06920*.

Ahmed, Z., Roux, N. L., Norouzi, M., and Schuurmans, D. (2018). Understanding the impact of entropy in policy learning. *arXiv preprint arXiv:1811.11214*.

Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Abbeel, O. P., and Zaremba, W. (2017). Hindsight experience replay. In *Advances in Neural Information Processing Systems*, pages 5048–5058.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. (2016). Openai gym. *arXiv preprint arXiv:1606.01540*.

Dayan, P. and Hinton, G. E. (1997). Using expectation-maximization for reinforcement learning. *Neural Computation*, 9(2):271–278.

Guadarrama, S., Korattikara, A., Ramirez, O., Castro, P., Holly, E., Fishman, S., Wang, K., Gonina, E., Harris, C., Vanhoucke, V., et al. (2018). Tf-agents: A library for reinforcement learning in tensorflow.

Gut, A. (2013). *Probability: a graduate course*, volume 75. Springer Science & Business Media.

Haarnoja, T., Pong, V., Zhou, A., Dalal, M., Abbeel, P., and Levine, S. (2018a). Composable deep reinforcement learning for robotic manipulation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6244–6251. IEEE.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. (2018b). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*.

Haarnoja, T., Zhou, A., Ha, S., Tan, J., Tucker, G., and Levine, S. (2018c). Learning to walk via deep reinforcement learning. *arXiv preprint arXiv:1812.11103*.

Heess, N., Wayne, G., Silver, D., Lillicrap, T., Erez, T., and Tassa, Y. (2015). Learning continuous control policies by stochastic value gradients. In *Advances in Neural Information Processing Systems*, pages 2944–2952.

Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B., and Madry, A. (2019). Adversarial examples are not bugs, they are features. *arXiv preprint arXiv:1905.02175*.

Kaelbling, L. P. (1993). Learning to achieve goals. In *IJCAI*, pages 1094–1099. Citeseer.

Kappen, H. J. (2005). Path integrals and symmetry breaking for optimal control theory. *Journal of statistical mechanics: theory and experiment*, 2005(11):P11011.

Levine, S. (2018). Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*.

Levine, S. and Koltun, V. (2013). Variational policy search via trajectory optimization. In *Advances in neural information processing systems*, pages 207–215.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. (2015). Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*.

Mihatsch, O. and Neuneier, R. (2002). Risk-sensitive reinforcement learning. *Machine learning*, 49(2-3):267–290.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540):529.

Neumann, G. et al. (2011). Variational inference for policy search in changing situations. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011*, pages 817–824.

Ng, A. Y. and Jordan, M. I. (2013). Pegasus: A policy search method for large mdps and pomdps. *arXiv preprint arXiv:1301.3878*.

Norouzi, M., Bengio, S., Jaitly, N., Schuster, M., Wu, Y., Schuurmans, D., et al. (2016). Reward augmented maximum likelihood for neural structured prediction. In *Advances In Neural Information Processing Systems*, pages 1723–1731.

Puterman, M. L. (2014). *Markov Decision Processes.: Discrete Stochastic Dynamic Programming*. John Wiley & Sons.

Rawlik, K., Toussaint, M., and Vijayakumar, S. (2013). On stochastic optimal control and reinforcement learning by approximate inference. In *Twenty-Third International Joint Conference on Artificial Intelligence*.

Schaul, T., Horgan, D., Gregor, K., and Silver, D. (2015). Universal value function approximators. In *International Conference on Machine Learning*, pages 1312–1320.

Simic, S. (2009). On an upper bound for jensen's inequality. *Journal of Inequalities in Pure and Applied Mathematics*, 10(2):5.

Singh, A., Yang, L., Hartikainen, K., Finn, C., and Levine, S. (2019). End-to-end robotic reinforcement learning without reward engineering. *arXiv preprint arXiv:1904.07854*.

Theodorou, E., Buchli, J., and Schaal, S. (2010). A generalized path integral control approach to reinforcement learning. *journal of machine learning research*, 11(Nov):3137–3181.

Thompson, W. R. (1933). On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294.

Todorov, E. (2007). Linearly-solvable markov decision problems. In *Advances in neural information processing systems*, pages 1369–1376.

Toussaint, M. (2009). Robot trajectory optimization using approximate inference. In *Proceedings of the 26th annual international conference on machine learning*, pages 1049–1056. ACM.

Williams, R. J. and Peng, J. (1991). Function optimization using connectionist reinforcement learning algorithms. *Connection Science*, 3(3):241–268.

Ziebart, B. D. (2010). *Modeling Purposeful Adaptive Behavior with the Principle of Maximum Causal Entropy*. PhD thesis, Carnegie Mellon University.