Balancing Two-Player Stochastic Games with Soft Q-Learning

Jordi Grau-Moya, Felix Leibfried and Haitham Bou-Ammar
PROWLER.io
jordi@prowler.io, felix@prowler.io, haitham@prowler.io

Abstract

Within the context of video games the notion of perfectly rational agents can be undesirable as it leads to uninteresting situations, where humans face tough adversarial decision makers. Current frameworks for stochastic games and reinforcement learning prohibit tuneable strategies as they seek optimal performance. In this paper, we enable such tuneable behaviour by generalising soft Q-learning to stochastic games, where more than one agent interact strategically. We contribute both theoretically and empirically. On the theory side, we show that games with soft Q-learning exhibit a unique value and generalise team games and zero-sum games far beyond these two extremes to cover a continuous spectrum of gaming behaviour. Experimentally, we show how tuning agents' constraints affect performance and demonstrate, through a neural network architecture, how to reliably balance games with high-dimensional representations.

1 Introduction

Stochastic Games (SG) provide a natural extension of reinforcement learning [Sutton and Barto, 1998; Mnih *et al.*, 2015; Busoniu *et al.*, 2010; Peters *et al.*, 2010] to multiple agents, where adapting strategies in presence of humans or other agents is necessary [Shapley, 1953; Littman, 1994; Littman, 2001].

Current frameworks for stochastic games assume perfectly rational agents – an assumption that is violated in a variety of real-world scenarios, e.g., human-robot interaction [Goodrich and Schultz, 2007], and pick-up and drop-off domains [Agussurja and Lau, 2012]. In the context of computer games, the main focus of this paper, such a problem of perfect rationality is even more amplified. Here, in fact, it is not desirable to design agents that seek optimal behaviour as this leads humans to become quickly uninterested when playing against adversarial agents that are impossible to defeat [Hunicke, 2005]. Hence, to design games with adaptable and balancing properties tailored to human-level performance, there is a need to extend state-of-the-art SG beyond optimality to allow for tuneable behaviour.

One method to produce tuneable behaviour in reinforcement learning is to introduce an adjustable Kullback-Leibler (KL) constraint between the agent's policy and a reference one. In particular, by increasingly strengthening this constraint we can obtain policies increasingly close to the reference policy, and vice versa. Bounding policy updates in such a manner has been previously introduced in literature under different names. Examples include KL control, relative entropy policy search [Peters *et al.*, 2010], path integral control [Kappen, 2005; Braun *et al.*, 2011], information-theoretic bounded rationality [Ortega and Braun, 2013], information-theory of decisions and actions [Tishby and Polani, 2011; Rubin *et al.*, 2012], and soft Q-learning [Fox *et al.*, 2016; Haarnoja *et al.*, 2017]. Targeted problems using these methods are also wide-spread, e.g., tackling the overestimation problem in tabular Q-learning [Fox *et al.*, 2016] and in Deep Q-networks [Leibfried *et al.*, 2017], accounting for model misspecification [Grau-Moya *et al.*, 2016], introducing safe policy updates in robot learning [Schulman *et al.*, 2015], and inducing risk-sensitive control [van den Broek *et al.*, 2010].

Contributions: Though abundant in literature, previous works only consider single-agent problems and are not readily applicable to stochastic games, which consider more than one interacting entity. With game balancing as our motivation, we propose a novel formulation of SG where agents are subject to KL constraints. In particular, our formulation introduces two KL constraints, one for each agent, limiting the space of available policies, which, in turn, enables tuneable behaviour. We then introduce an online strategy that can be used for game-play balancing even in high-dimensional spaces through a neural network architecture.

In short, the contributions of this paper can be summarised as: (1) proving convergence of the two-player soft Q-learning to a fixed point through contractions; (2) generalising team and zero-sum games in a continuous fashion and showing a unique value; (3) demonstrating convergence to correct behaviour by tuning the KL constraints on a simplified grid-world scenario; (4) extending our method to handle high-dimensional spaces; and (5) inferring opponent's Lagrange multiplier by maximum-likelihood, and demonstrating game-balancing behaviour on the game of Pong.

2 Background

2.1 Reinforcement Learning

In reinforcement learning (RL) [Sutton and Barto, 1998] an agent interacts with an unknown environment to determine an optimal policy that maximises total expected return. These problems are formalised as Markov decision processes (MDPs). Formally, an MDP is defined as the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma \rangle$ where $\mathcal{S}$ is the state space, $\mathcal{A}$ the action space, and $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ denotes the state transition density. Namely, when being in state $\boldsymbol{s}_t \in \mathcal{S}$ and applying an action $\boldsymbol{a}_t \in \mathcal{A}$, the agent transitions to $\boldsymbol{s}_{t+1} \sim \mathcal{T}(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{a}_t)$. The reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ quantifies the agent's performance and γ is the discount factor that trades off current and future rewards. The goal is to implement a policy that maximises total discounted rewards, i.e., $\pi^\star(\boldsymbol{a}|\boldsymbol{s}) = \operatorname{argmax}_\pi V^\pi(\boldsymbol{s})$, where $V^\pi(\boldsymbol{s}) = \mathbb{E}\left[\sum_{t=0}^\infty \gamma^t \mathcal{R}(\boldsymbol{s}_t, \boldsymbol{a}_t) \right]$.

2.2 Single Agent Soft Q-Learning

A way to constrain the behaviour of an agent is to modify the feasibility set of allowable policies. This can be achieved by introducing a constraint, such as a KL between two policy distributions, to the reinforcement learning objective. Such an approach has been already used within single agent reinforcement learning. For example, soft Q-learning has been used to reduce the overestimation problem of standard Q-learning [Fox *et al.*, 2016] and for building flexible energy-based policies in continuous domains [Haarnoja *et al.*, 2017]. Most of these approaches modify the standard objective of reinforcement learning to

$$\max_\pi \mathbb{E}\left[\sum_{t=0}^\infty \gamma^t \mathcal{R}(\boldsymbol{s}_t, \boldsymbol{a}_t) \right]$$
$$\text{s.t.} \sum_{t=0}^\infty \mathbb{E}\left[\gamma^t \text{KL}\left(\pi(\boldsymbol{a}_t|\boldsymbol{s}_t) || \rho(\boldsymbol{a}_t|\boldsymbol{s}_t) \right) \right] \le C, \quad (1)$$

where C is the amount of bits (or nats if using the natural logarithm) measured by the KL divergence that the policy π is allowed to deviate from a reference policy ρ. The expectation operation is over state-action trajectories.

To solve the above constrained problem, one typically introduces a Lagrange multiplier, β, and rewrites an equivalent unconstrained problem

$$\mathcal{V}^\star(\boldsymbol{s}) = \max_\pi \mathbb{E}\left[\sum_{t=0}^\infty \gamma^t \left(\mathcal{R}(\boldsymbol{s}_t, \boldsymbol{a}_t) - \frac{1}{\beta} \log \frac{\pi(\boldsymbol{a}_t|\boldsymbol{s}_t)}{\rho(\boldsymbol{a}_t|\boldsymbol{s}_t)} \right) \right].$$

To derive an algorithm for solving the above, one comes to recognise that $\mathcal{V}^\star(\boldsymbol{s})$ also satisfies a recursion similar to that introduced by the Bellman equations [Puterman, 1994]. Additionally, the optimal policy can be written in closed form as

$$\pi^\star(\boldsymbol{a}|\boldsymbol{s}) = \frac{\rho(\boldsymbol{a}|\boldsymbol{s})e^{\beta \mathcal{Q}^\star(\boldsymbol{s}, \boldsymbol{a})}}{\sum_{\boldsymbol{a} \in \mathcal{A}} \rho(\boldsymbol{a}|\boldsymbol{s})e^{\beta \mathcal{Q}^\star(\boldsymbol{s}, \boldsymbol{a})}},$$

where $\mathcal{Q}^\star(\boldsymbol{s}, \boldsymbol{a}) := \mathcal{R}(\boldsymbol{s}, \boldsymbol{a}) + \sum_{\boldsymbol{s}' \in \mathcal{S}} \mathcal{T}(\boldsymbol{s}'|\boldsymbol{s}, \boldsymbol{a})\mathcal{V}^\star(\boldsymbol{s}')$, and $\boldsymbol{s}' \in \mathcal{S}$. Notice that the above represents a generalisation of

standard RL settings, where $\beta \to \infty$ corresponds to a perfectly rational valuation ($\mathcal{V}^\star_{\beta \to \infty}(\boldsymbol{s}) = \max_\pi V^\pi(\boldsymbol{s})$), while for $\beta \to 0$ we recover the valuation under ρ ($\mathcal{V}^\star_{\beta \to 0}(\boldsymbol{s}) = V^\rho(\boldsymbol{s})$). Clearly, we can generate a continuum of policies between the reference and the perfectly rational policy that maximises the expected reward by tuning the choice of β as detailed in [Leibfried *et al.*, 2017].

2.3 Two-Player Stochastic Games

In two-player stochastic games [Shapley, 1953; Littman, 1994], two agents, that we denote as the player and the opponent, are interacting in an environment. Each agent executes a policy that we write as π_{pl} and π_{op}. At some time step t, the player chooses an action $\boldsymbol{a}_t^{\text{pl}} \sim \pi_{\text{pl}}(\boldsymbol{a}_t^{\text{pl}}|\boldsymbol{s}_t)$, while the opponent picks $\boldsymbol{a}_t^{\text{op}} \sim \pi_{\text{op}}(\boldsymbol{a}_t^{\text{op}}|\boldsymbol{s}_t)$. Accordingly, the environment transitions to a successor state $\boldsymbol{s}_{t+1} \sim \mathcal{T}_\mathcal{G}(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{a}_t^{\text{pl}}, \boldsymbol{a}_t^{\text{op}})$, where $\mathcal{T}_\mathcal{G}$ denotes the joint transition model for the game. After transitioning to a new state, both agents receive a particular reward depending on the type of game considered. In team games, both the player and the opponent maximise the same reward function $\mathcal{R}_\mathcal{G}(\boldsymbol{s}_t, \boldsymbol{a}_t^{\text{pl}}, \boldsymbol{a}_t^{\text{op}})$. For zero-sum games, the player seeks to maximise $\mathcal{R}_\mathcal{G}$, whereas the opponent seeks to find a minimum. We write the policy dependent value as $V^{\pi_{\text{pl}}\pi_{\text{op}}}(\boldsymbol{s}) := \mathbb{E}\left[\sum_{t=0}^\infty \gamma^t \mathcal{R}_\mathcal{G}(\boldsymbol{s}_t, \boldsymbol{a}_t^{\text{pl}}, \boldsymbol{a}_t^{\text{op}}) \right]$ where, in contrast to the one-player setting, the expectation is over state and *joint-action* trajectories.

In stochastic games it is common to assume perfect rationality for both agents i.e., in the case of a zero-sum game the player computes the optimal value of state $\boldsymbol{s}$ as $V^{\star\text{pl}}_{\text{zs}}(\boldsymbol{s}) = \max_{\pi_{\text{pl}}} \min_{\pi_{\text{op}}} V^{\pi_{\text{pl}}\pi_{\text{op}}}(\boldsymbol{s})$, while the opponent as $V^{\star\text{op}}_{\text{zs}}(\boldsymbol{s}) = \min_{\pi_{\text{op}}} \max_{\pi_{\text{pl}}} V^{\pi_{\text{pl}}\pi_{\text{op}}}(\boldsymbol{s})$. Similarly, in team games the optimal value for the player is $V^{\star\text{pl}}_{\text{tg}}(\boldsymbol{s}) = \max_{\pi_{\text{pl}}} \max_{\pi_{\text{op}}} V^{\pi_{\text{pl}}\pi_{\text{op}}}(\boldsymbol{s})$ and for the opponent $V^{\star\text{op}}_{\text{tg}}(\boldsymbol{s}) = \max_{\pi_{\text{opp}}} \max_{\pi_{\text{pl}}} V^{\pi_{\text{pl}}\pi_{\text{op}}}(\boldsymbol{s})$. Although it is straightforward to show that for team games $V^{\star\text{op}}_{\text{tg}}(\boldsymbol{s}) = V^{\star\text{pl}}_{\text{tg}}(\boldsymbol{s})$, an important classic result in game theory – the minimax theorem [Osborne and Rubinstein, 1994] – states that for zero-sum games $V^{\star\text{op}}_{\text{zs}}(\boldsymbol{s}) = V^{\star\text{pl}}_{\text{zs}}(\boldsymbol{s})$, i.e both team and zero sum games have a unique value.

Importantly, in complex games with large state-spaces the $\max$ and the $\min$ operations over all available policies are extremely difficult to compute. Humans and suboptimal agents seek to approximate these operations as best they can but never fully do so due to the lack of computational resources [Ortega and Stocker, 2016], approximations and introduced biases [Lieder *et al.*, 2012]. This limits the applicability of SG when interacting with suboptimal entities, e.g., in computer games when competing against human players. We next provide the first extension, to the best of our knowledge, of soft Q-learning to SGs and show how our framework can be used within the context of balancing the game's difficulty.

3 Two-Player Soft Q-Learning

To enable soft Q-learning in two-player games we introduce two KL constraints that allow us to separately control the per-

formance of both agents. In particular, we incorporate a constraint similar to (1) into the objective function for each agent and apply the method of Lagrange multipliers

$$\mathcal{V}^{\pi_{\mathrm{pl}}\pi_{\mathrm{op}}}(\boldsymbol{s}) = \mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t\left(\mathcal{R}_{\mathcal{G}}\left(\boldsymbol{s}_t, \boldsymbol{a}_t^{\mathrm{pl}}, \boldsymbol{a}_t^{\mathrm{op}}\right)\right.\right. \tag{2}$$
$$\left.\left. - \frac{1}{\beta_{\mathrm{pl}}}\log\frac{\pi_{\mathrm{pl}}(\boldsymbol{a}_t^{\mathrm{pl}}|\boldsymbol{s}_t)}{\rho_{\mathrm{pl}}(\boldsymbol{a}_t^{\mathrm{pl}}|\boldsymbol{s}_t)} - \frac{1}{\beta_{\mathrm{op}}}\log\frac{\pi_{\mathrm{op}}(\boldsymbol{a}_t^{\mathrm{op}}|\boldsymbol{s}_t)}{\rho_{\mathrm{op}}(\boldsymbol{a}_t^{\mathrm{op}}|\boldsymbol{s}_t)}\right)\right],$$

where the expectation is over joint-action trajectories, $\frac{1}{\beta_{\mathrm{pl}}}\log\frac{\pi_{\mathrm{pl}}(\boldsymbol{a}_t^{\mathrm{pl}}|\boldsymbol{s}_t)}{\rho_{\mathrm{pl}}(\boldsymbol{a}_t^{\mathrm{pl}}|\boldsymbol{s}_t)}$ is the information cost for the player (that turns into a KL divergence with the expectation operator), and $\frac{1}{\beta_{\mathrm{op}}}\log\frac{\pi_{\mathrm{op}}(\boldsymbol{a}_t^{\mathrm{op}}|\boldsymbol{s}_t)}{\rho_{\mathrm{op}}(\boldsymbol{a}_t^{\mathrm{op}}|\boldsymbol{s}_t)}$ is the information cost for the opponent. The Lagrange multipliers β_{pl} and β_{op} are tuneable parameters that we can vary at will. The distributions $\rho_{\mathrm{pl}}(\boldsymbol{a}_t^{\mathrm{pl}}|\boldsymbol{s}_t)$ and $\rho_{\mathrm{op}}(\boldsymbol{a}_t^{\mathrm{op}}|\boldsymbol{s}_t)$ are the arbitrary reference policies that we assume to be uniform[1]. Using the above, the player and the opponent compute optimal soft-value of a state $\boldsymbol{s}$ using

$$\mathcal{V}_{\mathrm{pl}}^{\star}(\boldsymbol{s}) = \max_{\pi_{\mathrm{pl}}}\mathop{\mathrm{ext}}_{\pi_{\mathrm{op}}}\mathcal{V}^{\pi_{\mathrm{pl}}\pi_{\mathrm{op}}}(\boldsymbol{s}),\ \mathcal{V}_{\mathrm{op}}^{\star}(\boldsymbol{s}) = \mathop{\mathrm{ext}}_{\pi_{\mathrm{op}}}\max_{\pi_{\mathrm{pl}}}\mathcal{V}^{\pi_{\mathrm{pl}}\pi_{\mathrm{op}}}(\boldsymbol{s}). \tag{3}$$

We define the extremum operator ext to correspond to a max in the case of positive β_{op} and to a min in the case of negative β_{op}.

It is clear that this novel formulation of the optimisation problems in Equations (3) generalise to cover both zero-sum and team games depending on the choice of β_{op}. By fixing $\beta_{\mathrm{pl}} \to \infty$ and setting $\beta_{\mathrm{op}} \to -\infty$ or $\beta_{\mathrm{op}} \to \infty$ we recover, respectively, a zero-sum or a team game with perfectly rational agents. For $\beta_{\mathrm{op}} \to 0$ we derive a game by which the opponent simply employs policy ρ_{op}. For finite values of β_{op}, we obtain a continuum of opponents with bounded performance ranging from fully adversarial to fully collaborative including a random policy. It is important to note, as we will show later, that the analytical form of the optimal policies that solve (3) are independent of the extremum operator and only depend on the parameters β_{pl} and β_{op}.

3.1 Unique Value for Two-Player Soft Q-Learning

In this section we show that the equations in (3) are equivalent, $\mathcal{V}_{\mathrm{pl}}^{\star}(\boldsymbol{s}) = \mathcal{V}_{\mathrm{op}}^{\star}(\boldsymbol{s})$, for any β_{pl} and β_{op}, that is, our two player soft Q-learning exhibit a unique value.

We start by defining the free energy operator as

$$f(\pi_{\mathrm{pl}}, \pi_{\mathrm{op}}, \boldsymbol{s}, \mathcal{V}) := \mathbb{E}_{\pi_{\mathrm{pl}},\pi_{\mathrm{op}}}\left[\mathcal{R}_{\mathcal{G}}(\boldsymbol{s}, \boldsymbol{a}^{\mathrm{pl}}, \boldsymbol{a}^{\mathrm{op}}) + \gamma\mathbb{E}_{\mathcal{T}_{\mathcal{G}}}\left[\mathcal{V}(\boldsymbol{s})\right]\right.$$
$$\left. - \frac{1}{\beta_{\mathrm{pl}}}\log\frac{\pi_{\mathrm{pl}}(\boldsymbol{a}^{\mathrm{pl}}|\boldsymbol{s})}{\rho_{\mathrm{pl}}(\boldsymbol{a}^{\mathrm{pl}}|\boldsymbol{s})} - \frac{1}{\beta_{\mathrm{opp}}}\log\frac{\pi_{\mathrm{op}}(\boldsymbol{a}^{\mathrm{op}}|\boldsymbol{s})}{\rho_{\mathrm{op}}(\boldsymbol{a}^{\mathrm{op}}|\boldsymbol{s})}\right] \tag{4}$$

for an arbitrary free energy vector $\mathcal{V}$. Then the Bellman-like operators for both the player and the opponent can be

expressed as:

$$\mathcal{B}_{\mathrm{pl}}\mathcal{V}(\boldsymbol{s}) = \max_{\pi_{\mathrm{pl}}}\mathop{\mathrm{ext}}_{\pi_{\mathrm{op}}}f(\pi_{\mathrm{pl}}, \pi_{\mathrm{op}}, \boldsymbol{s}, \mathcal{V}) \tag{5}$$
$$\mathcal{B}_{\mathrm{op}}\mathcal{V}(\boldsymbol{s}) = \mathop{\mathrm{ext}}_{\pi_{\mathrm{op}}}\max_{\pi_{\mathrm{pl}}}f(\pi_{\mathrm{pl}}, \pi_{\mathrm{op}}, \boldsymbol{s}, \mathcal{V}).$$

Proof Sketch: For proving our main results, summarised in Theorem 2, we commence by showing that the equations in (3) are equivalent. This is achieved by showing that the two operators in Equation (5) are in fact equivalent, see Lemma 1. Proving these operators to be contractions converging to a unique fixed point (see Theorem 1), we conclude that $\mathcal{V}_{\mathrm{pl}}^{\star}(\boldsymbol{s}) = \mathcal{V}_{\mathrm{op}}^{\star}(\boldsymbol{s})$.

Lemma 1. *For any $\beta_{pl} \in \mathbb{R}$ and $\beta_{op} \in \mathbb{R}$, and arbitrary free energy vector $\mathcal{V}$, then $\mathcal{B}_{pl}\mathcal{V}(\boldsymbol{s}) = \mathcal{B}_{op}\mathcal{V}(\boldsymbol{s})$.*

Due to Lemma 1, we can define the generic operator $\mathcal{B}\mathcal{V}(\boldsymbol{s}) := \mathcal{B}_{\mathrm{pl}}\mathcal{V}(\boldsymbol{s}) = \mathcal{B}_{\mathrm{op}}\mathcal{V}(\boldsymbol{s})$. Then, for this generic operator, we can prove the following.

Theorem 1 (Contraction). *For $\beta_{op} \in \mathbb{R}$ and $\beta_{pl} \in \mathbb{R}$, the operator $\mathcal{B}$ is an L_{∞}-norm contraction map $\|\mathcal{B}\mathcal{V} - \mathcal{B}\bar{\mathcal{V}}\|_{\infty} \leq \gamma\|\mathcal{V} - \bar{\mathcal{V}}\|_{\infty}$, where $\mathcal{V}$ and $\bar{\mathcal{V}}$ are two arbitrary free energy vectors and γ is the discount factor.*

Note that our reward is policy dependent (in the information cost) and, therefore, Theorem 1 is not a direct consequence of known results [Littman and Szepesvári, 1996], which assume that these rewards are policy independent. Using the above and the Banach's fixed point theorem [Puterman, 1994], we obtain the following corollary.

Corollary 1 (Unique fixed point). *The contraction mapping $\mathcal{B}$ exhibits a unique fixed-point $\mathcal{V}^{\star}$ such that $\mathcal{B}\mathcal{V}^{\star} = \mathcal{V}^{\star}$.*

Due to Lemma 1, Theorem 1 and Corollary 1, we arrive at the following.

Corollary 2. *Two-player stochastic games with soft Q-learning have a unique value, i.e. $\mathcal{V}_{pl}^{\star}(s) = \mathcal{V}_{op}^{\star}(s)$.*

3.2 Bounded-Optimal Policies

Corollary 2 allows us to exploit the fact that there exists one unique value to generate the policies for both agents. With this in mind, we next design an algorithm (similar in spirit to standard Q-Learning) that acquires tuneable policies. We start by defining a state-action value function, in resemblance to the Q-function, as

$$\mathcal{Q}^{\star}(\boldsymbol{s}, \boldsymbol{a}^{\mathrm{pl}}, \boldsymbol{a}^{\mathrm{op}}) := \mathcal{R}_{\mathcal{G}}(\boldsymbol{s}, \boldsymbol{a}^{\mathrm{pl}}, \boldsymbol{a}^{\mathrm{op}}) + \gamma\mathbb{E}_{\mathcal{T}_{\mathcal{G}}}\left[\mathcal{V}^{\star}(\boldsymbol{s}')\right].$$

For action selection, neither the player nor the opponent can directly use $\mathcal{Q}^{\star}$ as it depends on the action of the other agent, which is unknown a priori. Instead, it can be shown that agents must first compute the certainty equivalent by marginalising $\mathcal{Q}^{\star}$ as

$$\mathcal{Q}_{\mathrm{pl}}^{\star}(\boldsymbol{s}, \boldsymbol{a}^{\mathrm{pl}}) := \frac{1}{\beta_{\mathrm{op}}}\log\sum_{\boldsymbol{a}^{\mathrm{op}}}\rho_{\mathrm{op}}(\boldsymbol{a}^{\mathrm{op}}|\boldsymbol{s})\exp\left(\beta_{\mathrm{op}}\mathcal{Q}^{\star}(\boldsymbol{s}, \boldsymbol{a}^{\mathrm{pl}}, \boldsymbol{a}^{\mathrm{op}})\right)$$

$$\mathcal{Q}_{\mathrm{op}}^{\star}(\boldsymbol{s}, \boldsymbol{a}^{\mathrm{op}}) := \frac{1}{\beta_{\mathrm{pl}}}\log\sum_{\boldsymbol{a}^{\mathrm{pl}}}\rho_{\mathrm{pl}}(\boldsymbol{a}^{\mathrm{pl}}|\boldsymbol{s})\exp\left(\beta_{\mathrm{pl}}\mathcal{Q}^{\star}(\boldsymbol{s}, \boldsymbol{a}^{\mathrm{pl}}, \boldsymbol{a}^{\mathrm{op}})\right).$$

---

[1]Please note considering other reference policies is left as an interesting direction for future work.

With these definitions and using standard variational calculus, we obtain optimal policies for both the player and the opponent as[2]

$$\pi_{\mathrm{pl}}^{\star}(\boldsymbol{a}^{\mathrm{pl}}|\boldsymbol{s}) = \underset{\pi_{\mathrm{pl}}}{\operatorname{argmax}}\, \underset{\pi_{\mathrm{op}}}{\operatorname{ext}}\, f(\pi_{\mathrm{pl}}, \pi_{\mathrm{op}}, \boldsymbol{s}, \mathcal{V}^{\star})$$

$$= \frac{1}{Z_{\mathrm{pl}}(\boldsymbol{s})} \rho_{\mathrm{pl}}(\boldsymbol{a}^{\mathrm{pl}}|\boldsymbol{s}) \exp\left(\beta_{\mathrm{pl}} \mathcal{Q}_{\mathrm{pl}}^{\star}(\boldsymbol{s}, \boldsymbol{a}^{\mathrm{pl}})\right) \quad (6)$$

$$\pi_{\mathrm{op}}^{\star}(\boldsymbol{a}^{\mathrm{op}}|\boldsymbol{s}) = \underset{\pi_{\mathrm{op}}}{\operatorname{argext}}\, \underset{\pi_{\mathrm{pl}}}{\operatorname{max}}\, f(\pi_{\mathrm{pl}}, \pi_{\mathrm{op}}, \boldsymbol{s}, \mathcal{V}^{\star})$$

$$= \frac{1}{Z_{\mathrm{op}}(\boldsymbol{s})} \rho_{\mathrm{op}}(\boldsymbol{a}^{\mathrm{op}}|\boldsymbol{s}) \exp\left(\beta_{\mathrm{op}} \mathcal{Q}_{\mathrm{op}}^{\star}(\boldsymbol{s}, \boldsymbol{a}^{\mathrm{op}})\right),$$

where $Z_{\mathrm{pl}}(\boldsymbol{s})$ and $Z_{\mathrm{op}}(\boldsymbol{s})$ are normalising functions which can be exactly computed when assuming small discrete action spaces.

Hence, $\mathcal{V}^{\star}(\boldsymbol{s})$ can be expressed in closed form by incorporating the optimal policies in Equation (3) giving

$$\mathcal{V}^{\star}(\boldsymbol{s}) = \frac{1}{\beta_{\mathrm{pl}}} \log \sum_{\boldsymbol{a}^{\mathrm{pl}}} \rho_{\mathrm{pl}}(\boldsymbol{a}^{\mathrm{pl}}|\boldsymbol{s}) \exp\left(\beta_{\mathrm{pl}} \mathcal{Q}_{\mathrm{pl}}^{\star}(\boldsymbol{s}, \boldsymbol{a}^{\mathrm{pl}})\right). \quad (7)$$

As summarised in Algorithm 1, we learn $\mathcal{Q}^{\star}(\boldsymbol{s}, \boldsymbol{a}^{\mathrm{pl}}, \boldsymbol{a}^{\mathrm{op}})$ by applying the following recursion rule:

$$\mathcal{Q}_{k+1}(\boldsymbol{s}, \boldsymbol{a}^{\mathrm{pl}}, \boldsymbol{a}^{\mathrm{op}}) = \mathcal{Q}_k(\boldsymbol{s}, \boldsymbol{a}^{\mathrm{pl}}, \boldsymbol{a}^{\mathrm{op}}) \quad (8)$$
$$+ \alpha\big(\mathcal{R}_{\mathcal{G}}(\boldsymbol{s}, \boldsymbol{a}^{\mathrm{pl}}, \boldsymbol{a}^{\mathrm{op}}) + \gamma \mathcal{V}_k(\boldsymbol{s}') - \mathcal{Q}_k(\boldsymbol{s}, \boldsymbol{a}^{\mathrm{pl}}, \boldsymbol{a}^{\mathrm{op}})\big).$$

Here, α is the learning rate, k the learning step, and $\mathcal{V}_k(\boldsymbol{s}')$ is computed as in Equation (7) using the current $\mathcal{Q}_k$ estimate.

Algorithm 1 Two-Player Soft Q-Learning

1: Given $\rho_{\mathrm{pl}}, \rho_{\mathrm{op}}, \beta_{\mathrm{pl}}, \beta_{\mathrm{op}}, \mathcal{A}, \mathcal{S}$ and learning rate α
2: $\mathcal{Q}(\boldsymbol{s}, \boldsymbol{a}^{\mathrm{pl}}, \boldsymbol{a}^{\mathrm{op}}) \leftarrow 0$
3: **while** not converged **do**
4:   Collect transition $\left(\boldsymbol{s}_t, \boldsymbol{a}_t^{\mathrm{pl}}, \boldsymbol{a}_t^{\mathrm{op}}, \mathcal{R}_t, \boldsymbol{s}_t'\right)$, where $\boldsymbol{a}^{\mathrm{pl}} \sim \pi_{\mathrm{pl}}$, $\boldsymbol{a}^{\mathrm{op}} \sim \pi_{\mathrm{op}}$ and $\mathcal{R}_t$ is the reward at time t.
5:   Update $\mathcal{Q}$ according to Equation (8)
6: **end while**
7: **return** $\mathcal{Q}(\boldsymbol{s}, \boldsymbol{a}^{\mathrm{pl}}, \boldsymbol{a}^{\mathrm{op}})$

---

4 Real-World Considerations

Two restrictions limit the applicability of our algorithm to real-world scenarios. First, Algorithm 1 implicitly assumes the knowledge of the opponent's parameter β_{op}. Obtaining β_{op} in real-world settings can prove difficult. Second, our algorithm has been developed for low-dimensional state representations. Clearly, this restricts its applicability to high-dimensional states that are typical to computer games.

To overcome these issues, we next develop an online maximum likelihood procedure to infer β_{op} from data gathered through the interaction with the opponent, and then generalise Algorithm 1 to high-dimensional representations by proposing a deep learning architecture.

---

[2]Note that, if we assume that the action space has low cardinality, $\mathcal{Q}_{\mathrm{pl}}^{\star}(\boldsymbol{s}, \boldsymbol{a}^{\mathrm{pl}})$ and $\mathcal{Q}_{\mathrm{op}}^{\star}(\boldsymbol{s}, \boldsymbol{a}^{\mathrm{op}})$ can be computed exactly.

4.1 Estimating $\beta_{\mathbf{op}}$ & Game-Balancing

Rather than assuming access to the opponents rationality parameter, we next devise a maximum likelihood estimate that allows the agent to infer (in an online fashion) about β_{op} and consequently, about the real policy of the opponent (through Equation (6)).

Contrary to current SG techniques that attempt to approximate the opponent's policy directly, our method allows to reason about the opponent by only approximating a one dimensional parameter, i.e., β_{op} in Equation (6)[3].

Estimating $\beta_{\mathbf{op}}$: We frame the problem of estimating β_{op} as a one of online maximum likelihood estimation. Namely, we assume that the player interacts in R rounds with the opponent. At each round, j, the player gathers a dataset of the form $\mathcal{D}^{(j)} = \left\{\boldsymbol{s}_i^{(j)}, \boldsymbol{a}_i^{(j),\mathrm{pl}}, \boldsymbol{a}_i^{(j),\mathrm{op}}\right\}_{i=1}^{m^{(j)}}$ with $m^{(j)}$ denoting the total number of sampled transitions during round j. Given $\mathcal{D}^{(j)}$, the agent estimates its knowledge of the opponent's model i.e., β_{op} by solving the following problem[4]

$$\max_{\beta_{\mathrm{op}}} \log \mathrm{Pr}\left(\mathcal{D}^{(j)}\right) := \max_{\beta_{\mathrm{op}}} \sum_{i=1}^{m^{(j)}} \log \pi_{\mathrm{op}}^{\star}\left(\boldsymbol{a}_i^{(j),\mathrm{op}}|\boldsymbol{s}_i^{(j)}\right),$$

where $\pi_{\mathrm{op}}^{\star}\left(\boldsymbol{a}_i^{(j),\mathrm{op}}|\boldsymbol{s}_i^{(j)}\right)$ is defined in Equation (6). As rounds progress, the agent should learn to improve its estimate of β_{op}. Such an improvement is quantified, in terms of regret[5], in the following theorem for both a fixed and a time-varying opponent.

Theorem 2. *After R rounds, the average static-regret for estimating β_{op} vanishes as:*

$$\sum_{j=1}^{R} \mathcal{L}_j(\beta_{op}^{(j)}) - \min_{u}\left[\sum_{j=1}^{R} \mathcal{L}_j(u)\right] \approx \mathcal{O}(\sqrt{R}) \quad (9)$$

For a time-varying opponent, the dynamic regret bound dictates:

$$\sum_{j=1}^{R} \mathcal{L}_j(\beta_{op}^{(j)}) - \sum_{j=1}^{R} \min_{u_j} \mathcal{L}_j(u_j) \approx \mathcal{O}\Bigg(\sqrt{R}$$
$$\left(1 + \sum_{j=1}^{R-1} ||u_{j+1}^{\star} - u_j^{\star}||_2^2\right)\Bigg),$$

with $\mathcal{L}_j(\cdot)$ denoting the negative of the log-likelihood and $u_j^{\star} = \arg\min_u \mathcal{L}_j(u)$.

---

[3]Please note that similar to the previous section we assume the opponent's reference policy ρ_{op} to be uniform. This, however, does not impose a strong restriction since having a uniform reference policy enables enough flexibility to model various degrees of the opponent's performances (see Section 5).

[4]Please note that this problem can be easily solved using stochastic gradient descent.

[5]Regret is a standard notion to quantify the performance of an online learning algorithm. Regret measures the performance of the agent with respect to an adversary that has access to all information upfront.
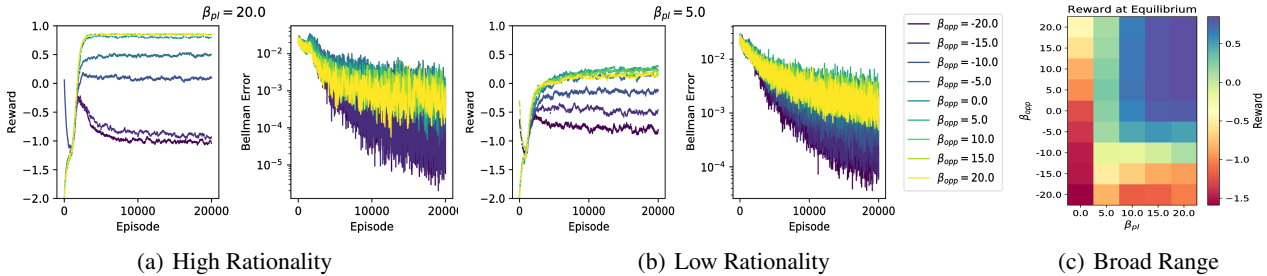
Figure 1: Evolution of reward and Bellman error during training for different β_{pl} and β_{op}. We vary β_{op} while fixing $\beta_{\text{pl}} = 20$ in panels (a), and $\beta_{\text{pl}} = 5.0$ in panels (b). The heat map in (c) visualises these rewards for a broader range of parameters. These results confirm that our approach can modulate performance.

From the above theorem we conclude that against a fixed-opponent our method guarantees correct approximation of β_{op}. This is true since the average regret, $\mathcal{O}(\sqrt{R})/R$, vanishes as $R \to \infty$. When it comes to a dynamic opponent, however, it is clear that our bound depends on how the value of the opponents multiplier parameter (in other words its policy) vary with in terms of rounds. In case these variations are bounded in number, we can still guarantee vanishing regrets. If not, the regret bound can grow arbitrarily large since $\sum_{j=1}^{R-1} ||u_{j+1}^\star - u_j^\star||_2^2$ can introduce a factor R.

Game Balancing: Now that we have a way to learn $\mathcal{Q}^\star$ and estimate β_{op} simultaneously, we could balance the game using the estimate of β_{op} to adjust the player's parameter β_{pl}. A simple heuristic that proved successful in our experiments was to simply set $\beta_{\text{pl}} = |\beta_{\text{op}}| + \Delta$, where Δ denotes an additional performance-level the player can achieve. Setting $\Delta = 0$ would correspond to agents with the same KL constraints, whereas setting $\Delta > 0$ would imply a stronger player with a softer KL constraint (see Section 5.2).

4.2 Deep Two-Player Soft Q-Learning

When tackling higher dimensional problems, one has to rely on function approximators to estimate the Q-function, or in our case, the function $\mathcal{Q}(\boldsymbol{s}, \boldsymbol{a}^{\text{pl}}, \boldsymbol{a}^{\text{op}})$. We borrow two ideas from deep Q-networks [Mnih *et al.*, 2015] that allow us to stabilise learning with high-dimensional representations for our SG setting. First, we use the notion of a replay memory to store the following transitions $(\boldsymbol{s}, \boldsymbol{a}^{\text{pl}}, \boldsymbol{a}^{\text{op}}, \boldsymbol{s}', \mathcal{R}_{\mathcal{G}}(\cdot))$ and, second, we use a target network denoted by $\mathcal{Q}(\boldsymbol{s}, \boldsymbol{a}^{\text{pl}}, \boldsymbol{a}^{\text{op}}; \boldsymbol{\theta}_i^-)$ to handle non-stationarity of the objective. We learn $\mathcal{Q}(\boldsymbol{s}, \boldsymbol{a}^{\text{pl}}, \boldsymbol{a}^{\text{op}}; \boldsymbol{\theta}_i)$, by using a neural network that receives $\boldsymbol{s}$ as input and outputs a matrix of $\mathcal{Q}$-values for each combination of the agents' actions. The loss function that we seek to minimise is $\mathcal{L}(\boldsymbol{\theta}_i) = \mathbb{E}\left[\left(\mathcal{R}_{\mathcal{G}}(\boldsymbol{s}, \boldsymbol{a}^{\text{pl}}, \boldsymbol{a}^{\text{op}}) + \gamma \mathcal{V}(\boldsymbol{s}; \boldsymbol{\theta}_i^-) - \mathcal{Q}(\boldsymbol{s}, \boldsymbol{a}^{\text{pl}}, \boldsymbol{a}^{\text{op}}; \boldsymbol{\theta}_i) \right)^2 \right]$ with the expectation taken over the distribution of transitions sampled from the replay memory, and $\mathcal{V}(\boldsymbol{s}; \boldsymbol{\theta}_i^-)$ computed as in Equation (7). Clearly, the above optimisation problem is similar to standard DQNs with the difference that error is measured between *soft Q-values*.

5 Experiments

We consider two cases in our experiments. The first assumes a low-dimensional setting, while the second targets the high-dimensional game of Pong. In both cases we consider full and no control of the opponent. Full control will allow us to validate our intuitions of tuneable behaviour, while the second sheds-the-light on the game balancing capabilities of Section 4.

5.1 Low-dimensional Experiments

The Setup: We validate Algorithm 1 on a 5×6 grid-world, where we consider two agents interacting. Each can choose an action from $\mathcal{A} = \{\text{left}, \text{right}, \text{up}, \text{down}, \text{pick-up}\}$. The first four actions are primitive movements, while the last corresponds to picking-up an object when possible. The reward of the first player is set to -0.02 for any movement and to $+1$ for picking-up the object located in cell $(2,6)$.

The setting described in this paper allows for a range of games that can be continuously varied between cooperative and defective games depending on the choice of β_{op} – a setting not allowed by any of the current techniques to stochastic games. In other words, the goal of the opponent, now, depends on the choice of β_{op}. Namely, for positive values of β_{op}, the opponent is collaborative, whereas for negative β_{op} it is adversarial. β_{op} values in between correspond to tuneable performance varying between the above two extremes.

We demonstrate adversarial behaviour by allowing agents to block each other either when trying to reach the same cell, or when attempting to transition to a cell previously occupied by the other agent. In such cases the respective agent remains in its current position. Given the determinism of the environment, a perfectly rational adversarial opponent can always impede the player to reach the goal. However, due to the KL constraints the opponent's policy becomes "less" aggressive, allowing the player to exploit the opponent's mistakes and arrive to the goal. For all experiments we used a high learning rate of $\alpha = 0.5$[6].

Tuning the Player's Performance: To validate tuneablity, we assess the performance of the player when reaching convergence while varying β_{pl} and β_{op}. In the first set of experiments, we fixed $\beta_{\text{pl}} = 20$ and varied $\beta_{\text{op}} = \{-20, -15, -10, -5, 0, 5, 10, 15, 20\}$. We expect that the player obtains high reward for collaborative opponents ($\beta_{\text{op}} > 0$) or highly sub-optimal adversarial opponents ($\beta_{\text{op}} \approx 0$), and low rewards for strong adversarial opponents

---

[6]A deterministic environment transitions allows for a large learning rate.

($\beta_{\mathrm{op}} \ll 0$). Indeed, the results shown in Figure 1(a) confirm these intuitions.

For a broader spectrum of analysis, we lower β_{pl} from 20 to 5 and re-run the same experiments. Results in Figure 1(b) reaffirm the previous conclusions. Here, however, the player attains slightly lower rewards as β_{pl} is decremented. Finally, in Figure 1(c) we plot the reward attained after convergence for a broad range of parameter values. We clearly see the effect of the modulation in both parameters on the resultant reward. The best reward is achieved when both parameters have positive high values, and the least reward for the lowest values.

Estimating $\beta_{\mathbf{op}}$: The goal of these experiments is to evaluate the correctness of our maximum likelihood estimate (Section 4.1) of β_{op}. To conduct these experiments, we fixed $\beta_{\mathrm{pl}} = 10$ and generated data with $\beta_{\mathrm{op}}^{\star} = 5$ and $\beta_{\mathrm{op}}^{\star} = -10$ that are unknown to the player. At each interaction with the environment, we updated $\mathcal{V}(\cdot)$ according to Algorithm 1 and β_{op} using a gradient step in the maximum likelihood objective. Results reported in Figure 2, clearly demonstrate that our extension to estimate β_{op} is successful[7].

5.2 High-dimensional Experiments

We repeat the experiments above but now considering our deep learning architecture of Section 4.2 on the game of Pong.

The Setup: We use the game Pong from the Roboschool package[8]. The state space is 13-dimensional i.e., x-y positions and x-y- velocities for both agents and the ball, and an additional dimension for time. We modified the action space to consist of $|\mathcal{A}| = 9$ actions where the set corresponds to $\mathcal{A} = \{\mathrm{left, stay, right}\} \times \{\mathrm{up, stay, down}\}$. We also modified the reward function to make it compatible with zero-sum games in such a way that if the player scores, the reward $\mathcal{R}_{\mathcal{G}}$ is set to $+1$, whereas if the opponent scores, to -1. The networks that represent soft Q-values, $\mathcal{Q}(\boldsymbol{s}, \boldsymbol{a}^{\mathrm{pl}}, \boldsymbol{a}^{\mathrm{op}})$, are multilayer perceptrons composed of two hidden layers, each with 100 units, and an a matrix output layer composed of $|\mathcal{A} \times \mathcal{A}| = 81$ units (9 × 9 actions). Here, each unit denotes a particular combination of $\boldsymbol{a}^{\mathrm{pl}} \in \mathcal{A}$ and $\boldsymbol{a}^{\mathrm{op}} \in \mathcal{A}$. After each hidden layer, we introduce a ReLU non-linearity. We used a learning rate of 10^{-4}, the ADAM optimizer, a batch size of 32, and updated the target every 30000 training steps.

Tuning the Player's Performance: In this experiment, we demonstrate successful tuneable performance. Figure 3 shows that for a highly adversarial opponent (i.e., $\beta_{\mathrm{op}} = -50$) the player ($\beta_{\mathrm{pl}} = 20$) acquired negative rewards, whereas for a weak opponent or even collaborative, the player obtained high reward. Game-play videos can be found at https://sites.google.com/site/submission3591/.

Estimating $\beta_{\mathbf{op}}$ and game balancing: Finally, we assess the performance of the maximum likelihood estimator applied to game balancing using neural networks. We pretrained a policy for the opponent with parameters $\beta_{\mathrm{pl}}^{\star} = 50.0$ and $\beta_{\mathrm{op}} = -20.0$, thus the player being stronger than the
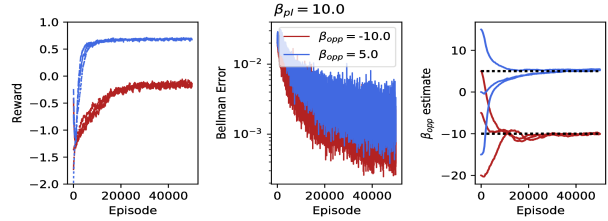

Figure 2: Reward, Bellman error and β_{op} estimate over episodes. We see that the maximum likelihood estimator is capable of discovering the correct value for the red ($\beta_{\mathrm{op}}^{\star} = -10$) and blue ($\beta_{\mathrm{op}}^{\star} = 5$) opponents, from three different initial estimates of β_{op}.
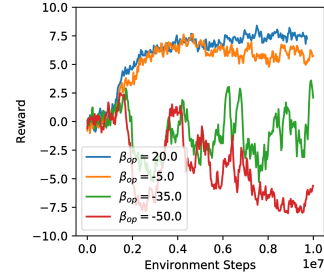
Figure 3: Results on Pong showing player's ($\beta_{\mathrm{pl}} = 20$) performance depending on β_{op}. We see that lower values of β_{op} yield more aggressive opponents, depicting lower reward for the player.
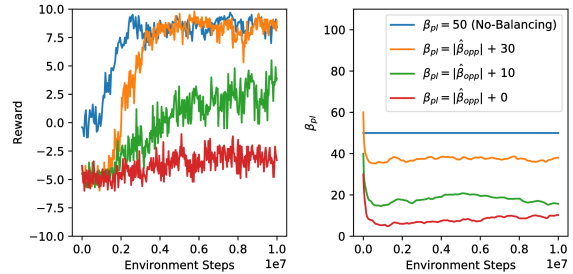

Figure 4: Player's performance depending on our game balancing scheme (see Section 4.1). We see that without balancing, the (fixed) player is much stronger than the opponent, whereas we obtain different performances depending on the balance parameter Δ. On the right, we show the parameter β_{pl} adapted online through the current $\hat{\beta}_{\mathrm{op}}$ estimate.

opponent (see blue line in Figure 4). In Figure 4, we demonstrate game balancing using Section 4.1. In particular, we are able to vary the player's performance by adapting (online) β_{pl}. For instance, if we set β_{pl} close to β_{op} we observe that the player is as strong as the opponent attaining 0 reward, see green line.

6 Conclusion

We extended two-player stochastic games to agents with KL constraints. We evaluated our method theoretically and empirically in both small and high-dimensional state spaces. The most interesting direction for future work is to scale our method to a large number of interacting agents by extending the approach in [Mguni *et al.*, 2018].

---

[7]In the case where the opponent would have an arbitrary policy π_a then β_{op} would converge to a value that attempts to make $\pi_{\mathrm{op}, \beta_{\mathrm{op}}}$ as close as possible to π_a.

[8]https://github.com/openai/roboschool

References

[Agussurja and Lau, 2012] Lucas Agussurja and Hoong Chuin Lau. Toward large-scale agent guidance in an urban taxi service. *Uncertainty in Artificial Intelligence*, 2012.

[Braun *et al.*, 2011] Daniel A Braun, Pedro A Ortega, Evangelos Theodorou, and Stefan Schaal. Path integral control and bounded rationality. In *Adaptive Dynamic Programming And Reinforcement Learning (ADPRL), 2011 IEEE Symposium on*, pages 202–209. IEEE, 2011.

[Busoniu *et al.*, 2010] Lucian Busoniu, Robert Babuska, Bart De Schutter, and Damien Ernst. *Reinforcement learning and dynamic programming using function approximators*, volume 39. CRC press, 2010.

[Fox *et al.*, 2016] Roy Fox, Ari Pakman, and Naftali Tishby. Taming the noise in reinforcement learning via soft updates. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*, pages 202–211. AUAI Press, 2016.

[Goodrich and Schultz, 2007] Michael A Goodrich and Alan C Schultz. Human-robot interaction: a survey. *Foundations and trends in human-computer interaction*, 1(3):203–275, 2007.

[Grau-Moya *et al.*, 2016] Jordi Grau-Moya, Felix Leibfried, Tim Genewein, and Daniel A Braun. Planning with information-processing constraints and model uncertainty in markov decision processes. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 475–491. Springer, 2016.

[Haarnoja *et al.*, 2017] Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *International Conference on Machine Learning*, pages 1352–1361, 2017.

[Hunicke, 2005] Robin Hunicke. The case for dynamic difficulty adjustment in games. In *Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology*, pages 429–433. ACM, 2005.

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

[Leibfried *et al.*, 2017] Felix Leibfried, Jordi Grau-Moya, and Haitham Bou-Ammar. An information-theoretic optimality principle for deep reinforcement learning. *arXiv preprint arXiv:1708.01867*, 2017.

[Lieder *et al.*, 2012] Falk Lieder, Tom Griffiths, and Noah Goodman. Burn-in, bias, and the rationality of anchoring. In *Advances in neural information processing systems*, pages 2690–2798, 2012.

[Littman and Szepesvári, 1996] Michael L Littman and Csaba Szepesvári. A generalized reinforcement-learning model: Convergence and applications. In *International Conference on Machine Learning*, 1996.

[Littman, 1994] Michael L Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the 11th International Conference on Machine Learning, 1994*, pages 157–163, 1994.

[Littman, 2001] Michael L Littman. Friend or foe q-learning in general-sum games. In *In Proceedings of the 18th Int. Conf. on Machine Learning*. Citeseer, 2001.

[Mguni *et al.*, 2018] David Mguni, Joel Jennings, and Enrique Munoz de Cote. Decentralised learning in systems with many, many strategic agents. In *AAAI Conference on Artificial Intelligence*, 2018.

[Mnih *et al.*, 2015] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[Ortega and Braun, 2013] Pedro A Ortega and Daniel A Braun. Thermodynamics as a theory of decision-making with information-processing costs. In *Proc. R. Soc. A*, volume 469, page 20120683. The Royal Society, 2013.

[Ortega and Stocker, 2016] Pedro A Ortega and Alan A Stocker. Human decision-making under limited time. In *Advances in Neural Information Processing Systems*, pages 100–108, 2016.

[Osborne and Rubinstein, 1994] Martin J Osborne and Ariel Rubinstein. *A course in game theory*. 1994.

[Peters *et al.*, 2010] Jan Peters, Katharina Mülling, and Yasemin Altün. Relative entropy policy search. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, pages 1607–1612. AAAI Press, 2010.

[Puterman, 1994] Martin Puterman. Markov decision processes: Discrete stochastic dynamic programming. 1994.

[Rubin *et al.*, 2012] Jonathan Rubin, Ohad Shamir, and Naftali Tishby. Trading value and information in mdps. *Decision Making with Imperfect Decision Makers*, pages 57–74, 2012.

[Schulman *et al.*, 2015] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 1889–1897, 2015.

[Shapley, 1953] Lloyd S Shapley. Stochastic games. *Proceedings of the national academy of sciences*, 39(10):1095–1100, 1953.

[Sutton and Barto, 1998] R Sutton and A Barto. *Reinforcement learning*. MIT Press, Cambridge, 1998.

[Tishby and Polani, 2011] Naftali Tishby and Daniel Polani. Information theory of decisions and actions. In *Perception-action cycle*, pages 601–636. Springer, 2011.

[van den Broek *et al.*, 2010] Bart van den Broek, Wim Wiegerinck, and Bert Kappen. Risk sensitive path integral control. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, UAI'10, pages 615–622, Arlington, Virginia, United States, 2010. AUAI Press.