On the Generalization Gap in Reparameterizable Reinforcement Learning

Huan Wang [1] **Stephan Zheng** [1] **Caiming Xiong** [1] **Richard Socher** [1]

Abstract

Understanding generalization in reinforcement learning (RL) is a significant challenge, as many common assumptions of traditional supervised learning theory do not apply. We focus on the special class of **reparameterizable RL** problems, where the trajectory distribution can be decomposed using the reparametrization trick. For this problem class, estimating the expected return is efficient and the trajectory can be computed deterministically given peripheral random variables, which enables us to study reparametrizable RL using supervised learning and transfer learning theory. Through these relationships, we derive guarantees on the gap between the expected and empirical return for both intrinsic and external errors, based on Rademacher complexity as well as the PAC-Bayes bound. Our bound suggests the generalization capability of reparameterizable RL is related to multiple factors including "smoothness" of the environment transition, reward and agent policy function class. We also empirically verify the relationship between the generalization gap and these factors through simulations.

1. Introduction

Reinforcement learning (RL) has proven successful in a series of applications such as games (Silver et al., 2016; 2017; Mnih et al., 2015; Vinyals et al., 2017; OpenAI, 2018), robotics (Kober et al., 2013), recommendation systems (Li et al., 2010; Shani et al., 2005), resource management (Mao et al., 2016; Mirhoseini et al., 2018), neural architecture design (Baker et al., 2017), and more. However some key questions in reinforcement learning remain unsolved. One that draws more and more attention is the issue of overfitting in reinforcement learning (Sutton, 1995; Cobbe et al., 2018; Zhang et al., 2018b; Packer et al., 2018; Zhang et al., 2018a). A model that performs well in the training environment, may or may not perform well when used in the testing environment. There is also a growing interest in understanding the conditions for model generalization and developing algorithms that improve generalization.

In general we would like to measure how accurate an algorithm is able to predict on previously unseen data. One metric of interest is the gap between the training and testing loss or reward. It has been observed that such gaps are related to multiple factors: initial state distribution, environment transition, the level of "difficulty" in the environment, model architectures, and optimization. Zhang et al. (2018b) split randomly sampled initial states into training and testing and evaluated the performance gap in deep reinforcement learning. They empirically observed overfitting caused by the randomness of the environment, even if the initial distribution and the transition in the testing environment are kept the same as training. On the other hand, Farebrother et al. (2018); Justesen et al. (2018); Cobbe et al. (2018) allowed the test environment to vary from training, and observed huge differences in testing performance. Packer et al. (2018) also reported very different testing behaviors across models and algorithms, even for the same RL problem.

Although overfitting has been empirically observed in RL from time to time, theoretical guarantees on generalization, especially finite-sample guarantees, are still missing. In this work, we focus on *on-policy RL*, where agent policies are trained based on episodes of experience that are sampled "on-the-fly" using the current policy in training. We identify two major obstacles in the analysis of on-policy RL. First, the episode distribution keeps changing as the policy gets updated during optimization. Therefore, episodes have to be continuously redrawn from the new distribution induced by the updated policy during optimization. For finite-sample analysis, this leads to a process with complex dependencies. Second, state-of-the-art research on RL tends to mix the errors caused by randomness in the environment and shifts in the environment distribution. We argue that actually these two types of errors are very different. One, which we call intrinsic error, is analogous to overfitting in supervised learning, and the other, called external error, looks more like the errors in transfer learning.

Our key observation is there exists a special class of RL,

[1]Salesforce Research, Palo Alto CA, USA. Correspondence to: Huan Wang <huan.wang@salesforce.com>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

called **reparameterizable RL**, where randomness in the environment can be decoupled from the transition and initialization procedures via the reparameterization trick (Kingma & Welling, 2014). Through reparameterization, an episode's dependency on the policy is "lifted" to the states. Hence, as the policy gets updated, episodes are deterministic given peripheral random variables. As a consequence, the expected reward in reparameterizable RL is connected to the Rademacher complexity as well as the PAC-Bayes bound. The reparameterization trick also makes the analysis for the second type of errors, i.e., when the environment distribution is shifted, much easier since the environment parameters are also "lifted" to the representation of states.

Related Work Generalization in reinforcement learning has been investigated a lot both theoretically and empirically. Theoretical work includes bandit analysis (Agarwal et al., 2014; Auer et al., 2002; 2009; Beygelzimer et al., 2011), Probably Approximately Correct (PAC) analysis (Jiang et al., 2017; Dann et al., 2017; Strehl et al., 2009; Lattimore & Hutter, 2014) as well as minimax analysis (Azar et al., 2017; Chakravorty & Hyland, 2003). Most works focus on the analysis of regret and consider the gap between the expected value and optimal return. On the empirical side, besides the previously mentioned work, Whiteson et al. (2011) proposes generalized methodologies that are based on multiple environments sampled from a distribution. Nair et al. (2015) also use random starts to test generalization.

Other research has also examined generalization from a transfer learning perspective. Lazaric (2012); Taylor & Stone (2009); Zhan & Taylor (2015); Laroche (2017) examine model generalization across different learning tasks, and provide guarantees on asymptotic performance.

There are also works in robotics for transferring policy from simulator to real world and optimizing an internal model from data (Kearns & Singh, 2002), or works trying to solve abstracted or compressed MDPs (Majeed & Hutter, 2018).

Our Contributions:

- A connection between (on-policy) reinforcement learning and supervised learning through the reparameterization trick. It simplifies the finite-sample analysis for RL, and yields Rademacher and PAC-Bayes bounds on Markov Decision Processes (MDP).
- Identifying a class of **reparameterizable RL** and providing a simple bound for "smooth" environments and models with a limited number of parameters.
- A guarantee for reparameterized RL when the environment is changed during testing. In particular we discuss two cases in environment shift: change in the initial distribution for the states, or the transition function.

2. Notation and Formulation

We denote a Markov Decision Process (MDP) as a 5-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \mathcal{P}_0)$. Here $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action-space, $\mathcal{P}(s, a, s') : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition probability from state s to s' when taking action a, $r(s) : \mathcal{S} \to \mathbb{R}$ represents the reward function, and $\mathcal{P}_0(s) : \mathcal{S} \to [0, 1]$ is the initial state distribution. Let $\pi(s) \in \Pi : \mathcal{S} \to \mathcal{A}$ be the policy map that returns the action a at state s.

We consider episodic MDPs with a finite horizon. Given the policy map π and the transition probability $\mathcal{P}$, the state-to-state transition probability is $\mathcal{T}_\pi(s, s') = \mathcal{P}(s, \pi(s), s')$. Without loss of generality, the length of the episode is $T + 1$. We denote a sequence of states $[s_0, s_1, \ldots, s_T]$ as $\boldsymbol{s}$. The total reward in an episode is $R(\boldsymbol{s}) = \sum_{t=0}^{T} \gamma^t r_t$, where $\gamma \in (0, 1]$ is a discount factor and $r_t = r(s_t)$.

Denote the joint distribution of the sequence of states in an episode $\boldsymbol{s} = [s_0, s_1, \ldots, s_T]$ as $\mathcal{D}_\pi$. Note $\mathcal{D}_\pi$ is also related to $\mathcal{P}$ and $\mathcal{P}_0$. In this work we assume $\mathcal{P}$ and $\mathcal{P}_0$ are fixed, so $\mathcal{D}_\pi$ is a function of π. Our goal is to find a policy that maximizes the expected total discounted reward (return):

$$\pi^* = \arg\max_{\pi \in \Pi} \mathbb{E}_{\boldsymbol{s} \sim \mathcal{D}_\pi} R(\boldsymbol{s}) = \arg\max_{\pi \in \Pi} \mathbb{E}_{\boldsymbol{s} \sim \mathcal{D}_\pi} \sum_{t=0}^{T} \gamma^t r_t. \tag{1}$$

Suppose during training we have a budget of n episodes, then the empirical return is

$$\hat{\pi} = \arg\max_{\pi \in \Pi, \boldsymbol{s}^i \sim \mathcal{D}_\pi} \frac{1}{n} \sum_{i=1}^{n} R(\boldsymbol{s}^i), \tag{2}$$

where $\boldsymbol{s}^i = [s_0^i, s_1^i, \ldots, s_T^i]$ is the ith episode of length $T + 1$. We are interested in the generalization gap

$$\Phi = \left| \frac{1}{n} \sum_{i=1}^{n} R(\boldsymbol{s}^i) - \mathbb{E}_{\boldsymbol{s} \sim \mathcal{D}'_{\hat{\pi}}} R(\boldsymbol{s}) \right|. \tag{3}$$

Note that in (3) the distribution $\mathcal{D}'_{\hat{\pi}}$ may be different from $\mathcal{D}_{\hat{\pi}}$ since in the testing environment $\mathcal{P}'$ as well as $\mathcal{P}'_0$ may be shifted compared to the training environment.

3. Generalization in Reinforcement Learning v.s. Supervised Learning

Generalization has been well studied in the supervised learning scenario. A popular assumption is that samples are independent and identically distributed $(x_i, y_i) \sim \mathcal{D}, \forall i \in \{1, 2, \ldots, n\}$. Similar to empirical return maximization discussed in Section 2, in supervised learning a popular algorithm is empirical risk minimization:

$$\hat{f} = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell(f, x_i, y_i), \tag{4}$$

where $f \in \mathcal{F} : \mathcal{X} \to \mathcal{Y}$ is the prediction function to be learned and $\ell : \mathcal{F} \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^+$ is the loss function. Similarly generalization in supervised learning concerns the gap between the expected loss $\mathbb{E}[\ell(f, x, y)]$ and the empirical loss $\frac{1}{n}\sum_{i=1}^{n} \ell(f, x_i, y_i)$.

It is easy to find the correspondence between the episodes defined in Section 2 and the samples (x_i, y_i) in supervised learning. Just like supervised learning where $(x, y) \sim \mathcal{D}$, in (episodic) reinforcement learning $\boldsymbol{s}^i \sim \mathcal{D}_\pi$. Also the reward function R in reinforcement learning is similar to the loss function ℓ in supervised learning. However, reinforcement learning is different because

- In supervised learning, the sample distribution $\mathcal{D}$ is kept fixed, and the loss function $\ell \circ f$ changes as we choose different predictors f.
- In reinforcement learning, the reward function R is kept fixed, but the sample distribution $\mathcal{D}_\pi$ changes as we choose different policy maps π.

As a consequence, the training procedure in reinforcement learning is also different. Popular methods such as REINFORCE (Williams, 1992), Q-learning (Sutton & Barto., 1998), and actor-critic methods (Mnih et al., 2016) draw new states and episodes on the fly as the policy π is being updated. That is, the distribution $\mathcal{D}_\pi$ from which episodes are drawn always changes during optimization. In contrast, in supervised learning we only update the predictor f without affecting the underlying sample distribution $\mathcal{D}$.

4. Intrinsic vs External Generalization Errors

The generalization gap (3) can be bounded

$$\Phi \le \underbrace{\left| \frac{1}{n}\sum_{i=1}^{n} R(\boldsymbol{s}^i) - \mathbb{E}_{\boldsymbol{s}\sim\mathcal{D}_{\hat{\pi}}} R(\boldsymbol{s}) \right|}_{\text{intrinsic}} + \underbrace{\left| \mathbb{E}_{\boldsymbol{s}\sim\mathcal{D}_{\hat{\pi}}} R(\boldsymbol{s}) - \mathbb{E}_{\boldsymbol{s}\sim\mathcal{D}'_{\hat{\pi}}} R(\boldsymbol{s}) \right|}_{\text{external}} \tag{5}$$

using the triangle inequality. The first term in (5) is the concentration error between the empirical reward and its expectation. Since it is caused by intrinsic randomness of the environment, we call it the *intrinsic error*. Even if the test environment shares the same distribution with training, in the finite-sample scenario there is still a gap between training and testing. This is analogous to the overfitting problem studied in supervised learning. Zhang et al. (2018b) mainly focuses on this aspect of generalization. In particular, their randomness is carefully controlled in experiments to only come from the initial states $s_0 \sim \mathcal{P}_0$.

We call the second term in (5) *external error*, as it is caused by shifts of the distribution in the environment. For example, the transition distribution $\mathcal{P}$ or the initialization distribution $\mathcal{P}_0$ may get changed during testing, which leads to a different underlying episode distribution $\mathcal{D}'_\pi$. This is analogous to the transfer learning problem. For instance, generalization as in Cobbe et al. (2018) is mostly external error since the number of levels used for training and testing are different even though the difficult level parameters are sampled from the same distribution. The setting in Packer et al. (2018) covers both intrinsic and external errors.

5. Why Intrinsic Generalization Error?

If π is fixed, by concentration of measures, as the number of episodes n increases, the intrinsic error decreases roughly with $\frac{1}{\sqrt{n}}$. For example, if the reward is bounded $|R(\boldsymbol{s}^i)| \le c/2$, by McDiarmid's bound, with probability at least $1-\delta$,

$$\left| \frac{1}{n}\sum_{i=1}^{n} R(\boldsymbol{s}^i) - \mathbb{E}_{\boldsymbol{s}\sim\mathcal{D}}[R(\boldsymbol{s})] \right| \le c\sqrt{\frac{\log\frac{2}{\delta}}{2n}}, \tag{6}$$

where $c > 0$. Note the bound above also holds for the test samples if the distribution $\mathcal{D}$ is fixed and $\boldsymbol{s}_{test} \sim \mathcal{D}$.

For the population argument (1), π^* is defined deterministically since the value $\mathbb{E}_{\boldsymbol{s}\sim\mathcal{D}_\pi} R(\boldsymbol{s})$ is a deterministic function of π. However, in the finite-sample case (2), the policy map $\hat{\pi}$ is stochastic: it depends on the samples $\boldsymbol{s}^i$. As a consequence, the underlying distribution $\mathcal{D}_{\hat{\pi}}$ is not fixed. In that case, the expectation $\mathbb{E}_{\boldsymbol{s}\sim\mathcal{D}_{\hat{\pi}}}[R(\boldsymbol{s})]$ in (6) becomes a random variable so (6) does not hold any more.

One way of fixing the issue caused by random $\mathcal{D}_{\hat{\pi}}$ is to prove a bound that holds uniformly for all policies $\pi \in \Pi$. If π is finite, by applying a union bound, it follows that:

Lemma 1. *If Π is finite, and $|R(\boldsymbol{s})| \le c/2$, then with probability at least $1-\delta$, for all $\pi \in \Pi$*

$$\left| \frac{1}{n}\sum_{i=1}^{n} R(\boldsymbol{s}^i) - \mathbb{E}_{\boldsymbol{s}\sim\mathcal{D}_\pi}[R(\boldsymbol{s})] \right| \le c\sqrt{\frac{\log\frac{2|\Pi|}{\delta}}{2n}}, \tag{7}$$

where $|\Pi|$ is the cardinality of Π.

Unfortunately in most of the applications, Π is not finite. One difficulty in analyzing the intrinsic generalization error is that the policy changes during the optimization procedure. This leads to a change in the episode distribution $\mathcal{D}_\pi$. Usually π is updated using episodes generated from some "previous" distributions, which are then used to generate new episodes. In this case it is not easy to split episodes into a training and testing set, since during optimization samples always come from the updated policy distribution.

Algorithm 1 Reparameterized MDP

Initialization: Sample $g_{init}, g_0, g_1, \ldots, g_T \sim \mathcal{G}^{|S|}$. $s_0 = \arg\max(g_{init} + \log \mathcal{P}_0)$, $R = 0$.
for t in $0, \ldots, T$ **do**
  $R = R + \gamma^t r(s_t)$
  $s_{t+1} = \arg\max(g_t + \log \mathcal{P}(s_t, \pi(s_t)))$
end for
return R.

6. Reparameterization Trick

The reparameterization trick has been popular in the optimization of deep networks (Kingma & Welling, 2014; Maddison et al., 2017; Jang et al., 2017; Tokui & Sato, 2016) and used, e.g., for the purpose of optimization efficiency. In RL, suppose the objective (1) is reparameterizable:

$$\mathbb{E}_{\boldsymbol{s}\sim\mathcal{D}_\pi} R(\boldsymbol{s}) = \mathbb{E}_{\xi\sim p(\xi)} R(\boldsymbol{s}(f(\xi, \pi))).$$

Then under some weak assumptions

$$\begin{aligned}\nabla_\theta \mathbb{E}_{\boldsymbol{s}\sim\mathcal{D}_{\pi_\theta}} R(\boldsymbol{s}) &= \nabla_\theta \left[\mathbb{E}_{\xi\sim p(\xi)} R(\boldsymbol{s}(f(\xi, \pi_\theta)))\right] \\ &= \mathbb{E}_{\xi\sim p(\xi)} \left[\nabla_\theta R(\boldsymbol{s}(f(\xi, \pi_\theta)))\right] \quad (8)\end{aligned}$$

The reparameterization trick has already been used: for example, PGPE (Rückstieß et al., 2010) uses policy reparameterization, and SVG (Heess et al., 2015) uses policy and environment dynamics reparameterization. In this work, we will show the reparameterization trick can help to analyze the generalization gap. More precisely, we will show that since both $\mathcal{P}$ and $\mathcal{P}_0$ are fixed, even if they are unknown, as long as they satisfy some "smoothness" assumptions, we can provide theoretical guarantees on the test performance.

7. Reparameterized MDP

We start our analysis with reparameterizing a Markov Decision Process with discrete states. We will give a general argument on reparameterizable RL in the next section. In this section we slightly abuse notation by letting $\mathcal{P}_0$ and $\mathcal{P}(s, a)$ denote $|S|$-dimensional probability vectors for multinomial distributions for initialization and transition respectively.

One difficulty in the analysis of the generalization in reinforcement learning rises from the sampling steps in MDP where states are drawn from multinomial distributions specified by either $\mathcal{P}_0$ or $\mathcal{P}(s_t, a_t)$, because the sampling procedure does not explicitly connect the states and the distribution parameters. We can use standard Gumbel random variables $g \sim \exp(-g + \exp(-g))$ to reparameterize sampling and get a procedure equivalent to classical MDPs but with slightly different expressions, as shown in Algorithm 1.

In the reparameterized MDP procedure, $\mathcal{G}^{|S|}$ is an $|S|$-dimensional Gumbel distribution. $g_0, \ldots, g_T$ are $|S|$-dimensional vectors with each entry being a Gumbel random variable. Also $g_0 + \log \mathcal{P}_0$ and $g_t + \log \mathcal{P}(s_t, a_t)$ are entry-wise vector sums, so they are both $|S|$-dimensional vectors. $\arg\max(v)$ returns the index of the maximum entry in the $|S|$-dimensional vector v. In the reparameterized MDP procedure shown above, the states s_t are represented as an index in $\{1, 2, \ldots, |S|\}$. After reparameterization, we may rewrite the RL objective (2) as:[1]

$$\hat{\pi} = \arg \max_{\pi\in\Pi, g^i\sim\mathcal{G}^{|S|T}} \frac{1}{n} \sum_{i=1}^{n} R(\boldsymbol{s}^i(g^i; \pi)), \quad (9)$$

where $g^i = [g_0^i, g_1^i, \ldots, g_T^i]$, g_t^i is an $|S|$-dimensional Gumbel random variable, and

$$R(\boldsymbol{s}^i(g^i; \pi)) = \sum_{t=0}^{T} \gamma^t r(s_t^i(g_0^i, g_1^i, \ldots, g_t^i; \pi)) \quad (10)$$

is the discounted return for one episode of length $T + 1$.

The reparameterized objective (9) maximizes the empirical reward by varying the policy π. The distribution from which the random variables g^i are drawn does not depend on the policy π anymore, and the policy π only affects the reward $R(\boldsymbol{s}^i(g^i; \pi))$ through the states $\boldsymbol{s}^i$.

The objective (9) is a discrete function due to the $\arg\max$ operator. One way to circumvent this is to use Gumbel softmax to approximate the $\arg\max$ operator (Maddison et al., 2017; Jang et al., 2017). If we denote s as a one-hot vector in $\mathbb{R}^{|S|}$, and further relax the entries in s to take positive values that sum up to one, we may use the softmax to approximate the $\arg\max$ operator. For instance, the reparametrized initial-state distribution becomes:

$$s_0 = \frac{\exp\{(g + \log \mathcal{P}_0)/\tau\}}{\|\exp\{(g + \log \mathcal{P}_0)/\tau\}\|_1}, \quad (11)$$

where g is an $|S|$-dimensional Gumbel random variable, $\mathcal{P}_0$ is an $|S|$-dimensional probability vector in multinomial distribution, and τ is a positive scalar. As the temperature $\tau \to 0$, the softmax approaches $s = \arg\max(g + \log \mathcal{P}_0) \sim \mathcal{P}_0$ in terms of the one-hot vector representation.

8. Reparameterizable RL

In general, as long as the transition and initialization process can be reparameterized so that the environment parameters are separated from the random variables, the objective can always be reformulated so that the policy only affects the reward instead of the underlying distribution. The **reparameterizable RL** procedure is shown in Algorithm 2.

[1] Again we abuse the notation by denoting $\boldsymbol{s}^i(f(g^i; \pi))$ as $\boldsymbol{s}^i(g^i; \pi)$.

Algorithm 2 Reparameterizzble RL

Initialization: Sample $\xi_0, \xi_1, \ldots, \xi_T$. $s_0 = \mathcal{I}(\xi_0)$, $R = 0$.
for t in $0, \ldots, T$ **do**
  $R = R + \gamma^t r(s_t)$
  $s_{t+1} = \mathcal{T}(s_t, \pi(s_t), \xi_t)$
end for
return R.

In this procedure, ξs are d-dimensional random variables but *they are not necessarily sampled from the same distribution*.[2] In many scenarios they are treated as random noise. $\mathcal{I} : \mathbb{R}^d \to \mathbb{R}^{|S|}$ is the *initialization function*. During initialization, the random variable ξ_0 is taken as input and the output is an initial state s_0. The transition function $\mathcal{T} : \mathbb{R}^{|S|} \times \mathbb{R}^{|A|} \times \mathbb{R}^d \to \mathbb{R}^{|S|}$, takes the current state s_t, the action produced by the policy $\pi(s_t)$, and a random variable ξ_t to produce the next state s_{t+1}.

In reparameterizable RL, the peripheral random variables $\xi_0, \xi_1, \ldots, \xi_T$ can be sampled before the episode is generated. In this way, the randomness is decoupled from the policy function, and as the policy π gets updated, the episodes can be computed deterministically.

The class of reparamterizable RL problems includes those whose initial state, transition, reward and optimal policy distribution can be reparameterized. Generally, a distribution can be reparameterized, e.g., if it has a tractable inverse CDF, is a composition of reparameterizable distributions (Kingma & Welling, 2014), or is a limit of smooth approximators (Maddison et al., 2017; Jang et al., 2017). Reparametrizable RL settings include LQR (Lewis et al., 1995) and physical systems (e.g., robotics) where the dynamics are given by stochastic partial differential equations (PDE) with reparameterizable components over continuous state-action spaces.

9. Main Result

For **reparameterizable RL**, if the environments and the policy are "smooth", we can control the error between the expected and the empirical reward. In particular, the assumptions we make are[3]

Assumption 1. *$\mathcal{T}(s,a) : \mathbb{R}^{|S|} \times \mathcal{R}^{|A|} \to \mathbb{R}^{|S|}$ is L_{t1}-Lipschitz in terms of the first variable s, and L_{t2}-Lipschitz in terms of the second variable a. That is, $\forall x, x', y, y', z$,*

$$\|\mathcal{T}(x,y,z) - \mathcal{T}(x',y,z)\| \le L_{t1}\|x - x'\|,$$
$$\|\mathcal{T}(x,y,z) - \mathcal{T}(x,y',z)\| \le L_{t2}\|y - y'\|.$$

[2] They may also have different dimensions. In this work, without loss of generality, we assume the random variables have the same dimension d.

[3] $\|\cdot\|$ is the L_2 norm, and $\theta \in \mathbb{R}^m$.

Assumption 2. *The policy is parameterized as $\pi(s;\theta) : \mathbb{R}^{|S|} \times \mathbb{R}^m \to \mathbb{R}^{|A|}$, and $\pi(s;\theta)$ is $L_{\pi 1}$-Lipschitz in terms of the states, and $L_{\pi 2}$-Lipschitz in terms of the parameter $\theta \in \mathbb{R}^m$, that is, $\forall s, s', \theta, \theta'$*

$$\|\pi(s;\theta) - \pi(s';\theta)\| \le L_{\pi 1}\|s - s'\|,$$
$$\|\pi(s;\theta) - \pi(s;\theta')\| \le L_{\pi 2}\|\theta - \theta'\|.$$

Assumption 3. *The reward $r(s) : \mathbb{R}^{|S|} \to \mathbb{R}$ is L_r-Lipschitz:*

$$|r(s') - r(s)| \le L_r\|s' - s\|.$$

If assumptions (1) (2) and (3) hold, we have the following:

Theorem 1. *In **reparameterizable RL**, suppose the transition $\mathcal{T}'$ in the test environment satisfies $\forall x, y, z, \|(\mathcal{T}' - \mathcal{T})(x,y,z)\| \le \zeta$, and suppose the initialization function $\mathcal{I}'$ in the test environment satisfies $\forall \xi, \|(\mathcal{I}' - \mathcal{I})(\xi)\| \le \epsilon$. If assumptions (1), (2) and (3) hold, the peripheral random variables ξ^i for each episode are i.i.d., and the reward is bounded $|R(\boldsymbol{s})| \le c/2$, then with probability at least $1 - \delta$, for all policies $\pi \in \Pi$:*

$$|\mathbb{E}_\xi[R(\boldsymbol{s}(\xi;\pi,\mathcal{T}',\mathcal{I}'))] - \frac{1}{n}\sum_i R(\boldsymbol{s}(\xi^i;\pi,\mathcal{T},\mathcal{I}))|$$
$$\le Rad(R_{\pi,\mathcal{T},\mathcal{I}}) + L_r\zeta\sum_{t=0}^{T}\gamma^t\frac{\nu^t - 1}{\nu - 1} + L_r\epsilon\sum_{t=0}^{T}\gamma^t\nu^t$$
$$+ O\left(c\sqrt{\frac{\log(1/\delta)}{n}}\right),$$

where $\nu = L_{t1} + L_{t2}L_{\pi 1}$, and $Rad(R_{\pi,\mathcal{T},\mathcal{I}}) = \mathbb{E}_\xi\mathbb{E}_\sigma\left[\sup_\pi \frac{1}{n}\sum_{i=1}^n \sigma_i R(\boldsymbol{s}^i(\xi^i;\pi,\mathcal{T},\mathcal{I}))\right]$ is the Rademacher complexity of $R(\boldsymbol{s}(\xi;\pi,\mathcal{T},\mathcal{I}))$ under the training transition $\mathcal{T}$, the training initialization $\mathcal{I}$, and n is the number if training episodes.

Note the i.i.d. assumption on the peripheral variables ξ^i is across episodes. Within the same episode, there could be correlations among the ξ_t^is at different time steps.

Similar arguments can also be made when the transition T' in the test environment stays the same as $\mathcal{T}$, but the initialization $\mathcal{I}'$ is different from $\mathcal{I}$. In the following sections we will bound the intrinsic and external errors respectively.

10. Bounding Intrinsic Generalization Error

After reparameterization, the objective (9) is essentially the same as an empirical risk minimization problem in the supervised learning scenario. According to classical learning theory, the following lemma is straight-forward (Shalev-Shwartz & Ben-David, 2014):

Lemma 2. *If the reward is bounded, $|R(\boldsymbol{s})| \leq c/2, c > 0$, and $g^i \sim G^{|S|\times T}$ are i.i.d. for each episode, with probability at least $1-\delta$, for $\forall \pi \in \Pi$:*

$$|\mathbb{E}_{g\sim\mathcal{G}^{|S|\times T}}[R(\boldsymbol{s}(g;\pi))] - \frac{1}{n}\sum_i R(\boldsymbol{s}^i(g^i;\pi))| \leq Rad(R_\pi) + O\left(c\sqrt{\frac{\log(1/\delta)}{n}}\right), \quad (12)$$

where $Rad(R_\pi) = \mathbb{E}_g\mathbb{E}_\sigma\left[\sup_\pi \frac{1}{n}\sum_{i=1}^n \sigma_i R(\boldsymbol{s}^i(g^i;\pi))\right]$ is the Rademacher complexity of $R(\boldsymbol{s}(g;\pi))$.

The bound (12) holds uniformly for all $\pi \in \Pi$, so it also holds for $\hat{\pi}$. Unfortunately, in MDPs $Rad(R_\pi)$ is hard to control, mainly due to the recursive $\arg\max$ in the representation of the states, $s_{t+1} = \arg\max(g_t + \log \mathcal{P}(s_t, \pi(s_t)))$.

On the other hand, for general reparameterizable RL we may control the intrinsic generalization gap by assuming some "smoothness" conditions on the transitions $\mathcal{T}$, as well as the policy π. In particular, it is straight-forward to prove that the empirical return R is "smooth" if the transitions and policies are all Lipschitz.

Lemma 3. *For **reparameterizable RL**, given assumptions 1, 2, and 3, the empirical return R defined in (10), as a function of the parameter θ, has a Lipschitz constant of*

$$\beta = L_r L_{t_2} L_{\pi 2} \sum_{t=0}^{T} \gamma^t \frac{\nu^t - 1}{\nu - 1}, \quad (13)$$

where $\nu = L_{t1} + L_{t2}L_{\pi 1}$.

Also, if the number of parameters m in $\pi(\theta)$ is bounded, then the Rademacher complexity $Rad(R_\pi)$ in Lemma 2 can be controlled (van der Vaart., 1998; Bartlett, 2013).

Lemma 4. *For **reparameterizable RL**, given assumptions 1, 2, and 3, if the parameters $\theta \in \mathbb{R}^m$ is bounded such that $\|\theta\| \leq 1$, and the function class of the reparameterized reward $\mathcal{R}$ is closed under negations, then the Rademacher complexity $Rad(R_\pi)$ is bounded by*

$$Rad(R_\pi) = O\left(\beta\sqrt{\frac{m}{n}}\right) \quad (14)$$

where β is the Lipschitz constant defined in (13), and n is the number of episodes.

In the context of deep learning, deep neural networks are over-parameterized models that have proven to work well in many applications. However, the bound above does not explain why over-parameterized models also generalize well since the Rademacher complexity bound (14) can be extremely large as m grows. To ameliorate this issue, recently Arora et al. (2018) proposed a compression approach that compresses a neural network to a smaller one with fewer parameters but has roughly the same training errors. Whether this also applies to reparameterizable RL is yet to be proven. There are also trajectory-based techniques proposed to sharpen the generalization bound (Li et al., 2018; Allen-Zhu et al., 2018; Arora et al., 2019; Cao & Gu, 2019).

10.1. PAC-Bayes Bound on Reparameterizable RL

We can also analyze the Rademacher complexity of the empirical return by making a slightly different assumption on the policy. Suppose π is parameterized as $\pi(\theta)$, and θ is sampled from some posterior distribution $\theta \sim \mathcal{Q}$. According to the PAC-Bayes theorem (McAllester, 1998; 2003; Neyshabur et al., 2018; Langford & Shawe-Taylor, 2002):

Lemma 5. *Given a "prior" distribution $\mathcal{D}_0$, with probability at least $1-\delta$ over the draw of n episodes, $\forall \mathcal{Q}$:*

$$\mathbb{E}_g[\mathrm{R}_{\theta\sim\mathcal{Q}}(g)] \geq \frac{1}{n}\sum_i \mathrm{R}_{\theta\sim\mathcal{Q}}(g^i) - 2\sqrt{\frac{2(KL(\mathcal{Q}||\mathcal{D}_0) + \log\frac{2n}{\delta})}{n-1}}, \quad (15)$$

$$R_{\theta\sim\mathcal{Q}}(g^i) = \mathbb{E}_{\theta\sim\mathcal{Q}}\left[R(\boldsymbol{s}^i(g^i;\pi(\theta)))\right] = \mathbb{E}_{\theta\sim\mathcal{Q}}\left[\sum_{t=0}^{T}\gamma^t r(s_t^i(g^i;\pi(\theta)))\right], \quad (16)$$

where $R_{\theta\sim\mathcal{Q}}(g)$ is the expected "Bayesian" reward.

The bound (15) holds for all posterior $\mathcal{Q}$. In particular it holds if $\mathcal{Q}$ is $\theta + u$ where θ could be any solution provided by empirical return maximization, and u is a perturbation, e.g., zero-centered uniform or Gaussian distribution. This suggests maximizing a perturbed objective instead may lead to better generalization performance, which has already been observed empirically (Wang et al., 2018b).

The tricky part about perturbing the policy is choosing the level of noise. Suppose there is an empirical reward optimizer $\pi(\hat{\theta})$. When the noise level is small, the first term in (15) is large, but the second term may also be large since the posterior $\mathcal{Q}$ is too focused on $\hat{\theta}$ but the "prior" $\mathcal{D}_0$ cannot depend on $\hat{\theta}$, and vice versa. On the other hand, if the reward function is "nice", e.g., if some "smoothness" assumption holds in a local neighborhood of $\hat{\theta}$, then one can prove the optimal noise level roughly scales inversely as the square root of the local Hessian diagonals (Wang et al., 2018a).

11. Bounding External Generalization Error

Another source of generalization error in RL comes from the change of environment. For example, in an MDP $(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \mathcal{P}_0)$, the transition probability $\mathcal{P}$ or the initialization distribution $\mathcal{P}_0$ is different in the test environment.

Cobbe et al. (2018) and Packer et al. (2018) show that as the distribution of the environment varies the gap between the training and testing could be huge.

Indeed if the test distribution is drastically different from the training environment, there is no guarantee the performance of the same model could possibly work for testing. On the other hand, if the test distribution $\mathcal{D}'$ is not too far away from the training distribution $\mathcal{D}$ then the test error can still be controlled. For example, for supervised learning, Mohri & Medina (2012) prove the expected loss of a drifting distribution is also bounded. In addition to Rademacher complexity and a concentration tail, there is one more term in the gap that measures the discrepancy between the training and testing distribution.

For reparameterizable RL, since the environment parameters are lifted into the reward function in the reformulated objective (9), the analysis becomes easier. For MDPs, a small change in environment could cause large difference in the reward since $\arg\max$ is not continuous. However if the transition function is "smooth", the expected reward in the new environment can also be controlled. e.g., if we assume the transition function $\mathcal{T}$, the reward function r, as well as the policy function π are all Lipschitz, as in section 10.

If the transition function $\mathcal{T}$ is the same in the test environment and the only difference is the initialization, we can prove the following lemma:

Lemma 6. *In **reparameterizable RL**, suppose the initialization function $\mathcal{I}'$ in the test environment satisfies $\forall \xi, \|(\mathcal{I}' - \mathcal{I})(\xi)\| \leq \zeta$ for $\zeta > 0$, and the transition function $\mathcal{T}$ in the test environment is the same as training. If assumptions (1), (2), and (3) hold, then:*

$$|\mathbb{E}_\xi[R(\boldsymbol{s}(\xi;\mathcal{I}'))] - \mathbb{E}_\xi[R(\boldsymbol{s}(\xi;\mathcal{I}))]| \leq L_r \zeta \sum_{t=0}^{T} \gamma^t (L_{t1} + L_{t2} L_{\pi 1})^t \tag{17}$$

Lemma 6 means that if the initialization in the test environment is not too different from the training one, and if the transition, policy and reward functions are smooth, then the expected reward in the test environment won't deviate from that of training too much.

The other possible environment change is that the test initialization $\mathcal{I}$ stays the same but the transition changes from the training transition $\mathcal{T}$ to $\mathcal{T}'$. Similar to before, we have:

Lemma 7. *In **reparameterizable RL**, suppose the transition $\mathcal{T}'$ in the test environment satisfies $\forall x, y, z, \|(\mathcal{T}' - \mathcal{T})(x, y, z)\| \leq \zeta$, and the initialization $\mathcal{I}$ in the test environment is the same as training. If assumptions (1), (2) and (3) hold then*

$$|\mathbb{E}_\xi[R(\boldsymbol{s}(\xi;\mathcal{T}'))] - \mathbb{E}_\xi[R(\boldsymbol{s}(\xi;\mathcal{T}))]| \leq L_r \zeta \sum_{t=0}^{T} \gamma^t \frac{\nu^t - 1}{\nu - 1} \tag{18}$$

where $\nu = L_{t1} + L_{t2} L_{\pi 1}$.

The difference between (18) and (17) is that the change ζ in transition $\mathcal{T}$ is further enlarged during an episode: as long as $\nu > 1$, the gap in (18) is larger and can become huge as the length T of the episode increases.

Table 1. Intrinsic Gap versus Smoothness

Temperature	**Policy**		**State**	**Action**
τ	Gap	$\frac{1}{\tau}\Pi_l \|\hat{\theta}^l\|_F$	Gap	Gap
0.001	0.554	$2.20 \cdot 10^6$	0.632	0.612
0.01	0.494	$4.46 \cdot 10^5$	0.632	0.608
0.1	0.482	$1.74 \cdot 10^5$	0.633	0.603
1	0.478	$8.83 \cdot 10^4$	0.598	0.598
10	0.479	$5.06 \cdot 10^4$	0.588	0.594
100	0.468	$4.77 \cdot 10^4$	0.581	0.594
1000	0.471	$3.29 \cdot 10^4$	0.590	0.594

12. Simulation

We now present empirical measurements in simulations to verify some claims made in section 10 and 11. The bound (14) suggests the gap between the expected reward and the empirical reward is related to the Lipschitz constant β of R, which according to equation (13) is related to the Lipschitz constant of a series of functions including π, $\mathcal{T}$, and r.

12.1. Intrinsic Generalization Gap

In (13), as the length of the episode T increases, the dominating factors in β are L_{t1}, L_{t2} and $L_{\pi 1}$. Our first simulation fixes the environment and verifies L_π. In the simulation, we assume the initialization $\mathcal{I}$ and the transition $\mathcal{T}$ are all known and fixed. $\mathcal{I}$ is an identity function, and $\xi_0 \in \mathbb{R}^{|S|}$ is a vector of i.i.d. uniformly distributed random variables: $\xi_0[k] \sim U[0,1], \forall k \in 1, \dots |S|$. The transit function is $\mathcal{T}(s, a, \xi) = sT_1 + aT_2 + \xi T_3$, where $s \in \mathbb{R}^{|S|}$, $a \in \mathbb{R}^{|A|}$, $\xi \in \mathbb{R}^2$ are row vectors, and $T_1 \in \mathbb{R}^{|S| \times |S|}$, $T_2 \in \mathbb{R}^{|A| \times |S|}$, and $T_3 \in \mathbb{R}^{2 \times |S|}$ are matrices used to project the states, actions, and noise respectively. T_1, T_2, and T_3 are randomly generated and then kept fixed during the experiment. We use $\gamma = 1$ as the discounting constant throughout.

The policy $\pi(s, \theta)$ is modeled using a multiple layer perceptron (MLP) with rectified linear as the activation. The last layer of MLP is a linear layer followed by a softmax function with temperature: $q(x[k]; \tau) = \frac{\exp \frac{x[k]}{\tau}}{\sum_k \exp \frac{x[k]}{\tau}}$.

By varying the temperature τ we are able to control the Lipschitz constant of the policy class $L_{\pi 1}$ and $L_{\pi 2}$ if we assume the bound on the parameters $\|\theta\| \leq B$ is unchanged.

We set the length of the episode $T = 128$, and randomly sample $\xi_0, \xi_1, \ldots, \xi_T$ for $n = 128$ training and testing episodes. Then we use the same random noise to evaluate a series of policy classes with different temperatures $\tau \in \{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$.

Since we assume $\mathcal{I}$ and $\mathcal{T}$ are known, during training the computation graph is complete. Hence we can directly optimize the coefficients θ in $\pi(s;\theta)$ just as in supervised learning.[4] We use Adam (Kingma & Ba, 2015) to optimize with initial learning rates 10^{-2} and 10^{-3}. When the reward stops increasing we halved the learning rate. and analyze the gap between the average training and testing reward.

First, we observe the gap is affected by the optimization procedure. For example, different learning rates can lead to different local optima, even if we decrease the learning rate by half when the reward does not increase. Second, even if we know the environment $\mathcal{I}$ and $\mathcal{T}$, so that we can optimize the policy $\pi(s;\theta)$ directly, we still experience unstable learning just like other RL algorithms. This suggests that the unstableness of the RL algorithms may not rise from the estimation of the environment for the model based algorithms such as A2C and A3C (Mnih et al., 2016), since even if we know the environment the learning is still unstable.

Given the unstable training procedure, for each trial we ran the training for 1024 epochs with learning rate of 1e-2 and 1e-3, and the one with higher training reward at the last epoch is used for reporting. Ideally as we vary τ, the Lipschitz constant for the function class $\pi \in \Pi$ is changed accordingly given the assumption $\|\theta\| \leq B$. However, it is unclear if B is changed or not for different configurations. After all, the assumption that the parameters are bounded is artificial. To ameliorate this defect we also check the metric $\frac{1}{\tau}\Pi_l \|\theta^l\|_F$, where θ^l is the weight matrix of the lth layer of MLP. In our experiment there is no bias term in the linear layers in MLP, so $\frac{1}{\tau}\Pi_l \|\hat{\theta}^l\|_F$ can be used as a metric on the Lipschitz constant $L_{\pi 1}$ at the solution point $\hat{\theta}$. We also vary the smoothness in the transition function a a function of states (T_1), and actions (T_2), by applying softmax with different temperatures τ to the singular values of the randomly generated matrix.

Table 1 shows the average generalization gap roughly decreases as τ decreases. The metric $\frac{1}{\tau}\Pi_l \|\hat{\theta}^l\|_F$ also decreases similarly as the average gap. In particular, the 2nd and 3rd column shows the average gap as the policy becomes "smoother". The 4th column shows, if we fix the policy-τ as well as setting $T_2 = 1$, the generalization gap decreases as we increase the transition-τ for T_1 (states). Similarly the last column is the gap as the transition-τ for actions (T_2) varies. In Table 2 the environment is fixed and for each parameter configuration the gap is averaged from 100 trials with randomly initialized and then optimized policies.

[4] In real applications this is not doable since $\mathcal{T}$ and $\mathcal{I}$ are unknown. Here we assume they are known just to investigate the generalization gap.

Params	65.6k	131.3k	263.2k	583.4k	1.1m
Gap	0.204	0.183	0.214	0.336	0.418

Table 2. Empirical gap vs #policy params.

ζ **in** $\mathcal{I}$	1	10	100	1,000
Gap	0.481	0.477	0.659	0.532

Table 3. Empirical generalization gap vs shift in initialization.

ζ **in** $\mathcal{T}$	1	10	100	1,000
Gap	11	451	8,260	73,300

Table 4. Empirical generalization gap vs shift in transition.

12.2. External Generalization Gap

To measure the external generalization gap, we vary the transition $\mathcal{T}$ as well as the initialization $\mathcal{I}$ in the test environment. For that, we add a vector of Rademacher random variables Δ to $\mathcal{I}$ or $\mathcal{T}$, with $\|\Delta\| = \zeta$. We adjust the level of noise δ in the simulation and report the change of the average gap in Table 3 and Table 4. It is not surprising that the change Δ_T in transition $\mathcal{T}$ leads to a higher generalization gap since the impact from Δ_T is accumulated across time steps. Indeed if we compare the bound (18) and (17), when $\gamma = 1$ as long as $\nu > 1$, the gap in (18) is larger.

13. Discussion and Future Work

Even though a variety of distributions, discrete or continuous, can be reparameterized, and we have shown that the classical MDP with discrete states is reparameterizable, it is not clear in general under which conditions reinforcement learning problems are reparameterizable. Classifying particular cases where RL is not reparameterizable is an interesting direction for future work. Second, the transitions of discrete MDPs are inherently non-smooth, so Theorem 1 does not apply. In this case, the PAC-Bayes bound can be applied, but this requires a totally different framework. It will be interesting to see if there is a "Bayesian" version of Theorem 1. Finally, our analysis only covers "on-policy" RL. Studying generalization for "off-policy" RL remains an interesting future topic.

References

Agarwal, A., Hsu, D., Kale, S., Langford, J., Li, L., and Schapire, R. Taming the monster: A fast and simple al-

gorithm for contextual bandits. *International Conference on Machine Learning*, 2014.

Allen-Zhu, Z., Li, Y., and Liang, Y. Learning and generalization in overparameterized neural networks, going beyond two layers. *CoRR*, abs/1811.04918, 2018.

Arora, S., Ge, R., Neyshabur, B., and Zhang, Y. Stronger generalization bounds for deep nets via a compression approach. *International Conference on Machine Learning*, 2018.

Arora, S., Du, S. S., Hu, W., Li, Z., and Wang, R. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. *International Conference on Machine Learning*, 2019.

Auer, P., Cesa-Bianchi, N., and Fischer, P. Finite-time analysis of the multiarmed bandit problem. *Maching Learning*, 2002.

Auer, P., Jaksch, T., and Ortner, R. Near-optimal regret bounds for reinforcement learning. *Advances in Neural Information Processing Systems 21*, 2009.

Azar, M. G., Osband, I., and Munos, R. Minimax regret bounds for reinforcement learning. *International Conference on Machine Learning*, 2017.

Baker, B., Gupta, O., Naik, N., and Raskar, R. Designing neural network architectures using reinforcement learning. 2017.

Bartlett, P. Lecture notes on theoretical statistics. 2013.

Beygelzimer, A., Langford, J., Li, L., Reyzin, L., and Schapire, R. Contextual bandit algorithms with supervised learning guarantees. *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011.

Cao, Y. and Gu, Q. A generalization theory of gradient descent for learning over-parameterized deep relu networks. *CoRR*, abs/1902.01384, 2019.

Chakravorty, S. and Hyland, D. C. Minimax reinforcement learning. *American Institute of Aeronautics and Astronautic*, 2003.

Cobbe, K., Klimov, O., Hesse, C., Kim, T., and Schulman, J. Quantifying generalization in reinforcement learning. *CoRR*, 2018. URL http://arxiv.org/abs/1812.02341.

Dann, C., Lattimore, T., and Brunskill, E. Unifying pac and regret: Uniform pac bounds for episodic reinforcement learning. *International Conference on Neural Information Processing Systems (NIPS)*, 2017.

Farebrother, J., Machado, M. C., and Bowling, M. Generalization and regularization in dqn. *CoRR*, 2018. URL https://arxiv.org/abs/1810.00123.

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

Jang, E., Gu, S., and Poole, B. Categorical reparameterization by gumbel-softmax. *International Conference on Learning Representations*, 2017.

Jiang, N., Krishnamurthy, A., Agarwal, A., Langford, J., and Schapire, R. E. Contextual decision processes with low Bellman rank are PAC-learnable. *International Conference on Machine Learning*, 2017.

Justesen, N., Torrado, R. R., Bontrager, P., Khalifa, A., Togelius, J., and Risi, S. Illuminating generalization in deep reinforcement learning through procedural level generation. *NeurIPS Deep RL Workshop*, 2018.

Kearns, M. and Singh, S. Near-optimal reinforcement learning in polynomial time. *Mache Learning*, 2002.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 2015.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *International Conference on Learning Representations*, 2014.

Kober, J., Bagnell, J. A., and Peters, J. Reinforcement learning in robotics: A survey. *International Journal of Robotic Research*, 2013.

Langford, J. and Shawe-Taylor, J. Pac-bayes & margins. *International Conference on Neural Information Processing Systems (NIPS)*, 2002.

Laroche, R. Transfer reinforcement learning with shared dynamics. 2017.

Lattimore, T. and Hutter, M. Near-optimal PAC bounds for discounted mdps. *Theoretical Computer Science*, 2014.

Lazaric, A. Transfer in reinforcement learning: a framework and a survey. *Reinforcement Learning - State of the Art, Springer*, 2012.

Lewis, F., Syrmos, V., and Syrmos, V. *Optimal Control*. A Wiley-interscience publication. Wiley, 1995. ISBN 9780471033783. URL https://books.google.com/books?id=jkD37elP6NIC.

Li, L., Chu, W., Langford, J., and Schapire, R. E. A contextual-bandit approach to personalized news article recommendation. *Proceedings of the 19th International Conference on World Wide Web*, 2010.

Li, Y., Ma, T., and Zhang, H. Algorithmic regularization in over-parameterized matrix recovery, 2018.

Maddison, C. J., Mnih, A., and Teh, Y. W. The concrete distribution: a continuous relaxation of discrete random variables. *International Conference on Learning Representations*, 2017.

Majeed, S. J. and Hutter, M. Performance guarantees for homomorphisms beyond markov decision processes. *CoRR*, abs/1811.03895, 2018.

Mao, H., Alizadeh, M., Menache, I., and Kandula, S. Resource management with deep reinforcement learning. 2016.

McAllester, D. A. Some pac-bayesian theorems. *Conference on Learning Theory (COLT)*, 1998.

McAllester, D. A. Simplified pac-bayesian margin bounds. *Conference on Learning Theory (COLT)*, 2003.

Mirhoseini, A., Goldie, A., Pham, H., Steiner, B., Le, Q. V., and Dean, J. Hierarchical planning for device placement. 2018. URL https://openreview.net/pdf?id=Hkc-TeZ0W.

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. Human-level control through deep reinforcement learning. *Nature*, 2015.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. *International Conference on Machine Learning*, 2016.

Mohri, M. and Medina, A. M. New analysis and algorithm for learning with drifting distributions. *Algorithmic Learning Theory*, 2012.

Nair, A., Srinivasan, P., Blackwell, S., Alcicek, C., Fearon, R., Maria, A. D., Panneershelvam, V., Suleyman, M., Beattie, C., Petersen, S., Legg, S., Mnih, V., Kavukcuoglu, K., and Silver, D. Massively parallel methods for deep reinforcement learning. *CoRR*, abs/1507.04296, 2015. URL http://arxiv.org/abs/1507.04296.

Neyshabur, B., Bhojanapalli, S., and Srebro, N. A pac-bayesian approach to spectrally-normalized margin bounds for neural networks. *International Conference on Learning Representations (ICLR)*, 2018.

OpenAI. Openai five. https://blog.openai.com/openai-five/, 2018.

Packer, C., Gao, K., Kos, J., Krähenbühl, P., Koltun, V., and Song, D. Assessing generalization in deep reinforcement learning. *CoRR*, 2018. URL https://arxiv.org/abs/1810.12282.

Rückstieß, T., Sehnke, F., Schaul, T., Wierstra, D., Sun, Y., and Schmidhuber, J. Exploring parameter space in reinforcement learning. *Paladyn*, 2010.

Shalev-Shwartz, S. and Ben-David, S. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, New York, NY, USA, 2014. ISBN 1107057132, 9781107057135.

Shani, G., Brafman, R. I., and Heckerman, D. An mdp-based recommender system. *The Journal of Machine Learning Research*, 2005.

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. Mastering the game of go with deep neural networks and tree search. *Nature*, 2016.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T. P., Simonyan, K., and Hassabis, D. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *CoRR*, 2017. URL http://arxiv.org/abs/1712.01815.

Strehl, A. L., Li, L., and Littman, M. L. Reinforcement learning in finite mdps: Pac analysis. *Journal of Machine Learning Research*, 2009.

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

Sutton, R. S. Generalization in reinforcement learning: Successful examples using sparse coarse coding. 1995.

Taylor, M. E. and Stone, P. Transfer learning for reinforcement learning domains: A survey. *J. Mach. Learn. Res.*, 2009.

Tokui, S. and Sato, I. Reparameterization trick for discrete variables. *CoRR*, 2016. URL https://arxiv.org/abs/1611.01239.

van der Vaart., A. *Asymptotic Statistics.*. Cambridge, 1998.

Vinyals, O., Ewalds, T., Bartunov, S., Georgiev, P., Vezhnevets, A. S., Yeo, M., Makhzani, A., Küttler, H., Agapiou, J., Schrittwieser, J., Quan, J., Gaffney, S., Petersen,

S., Simonyan, K., Schaul, T., van Hasselt, H., Silver, D., Lillicrap, T. P., Calderone, K., Keet, P., Brunasso, A., Lawrence, D., Ekermo, A., Repp, J., and Tsing, R. Starcraft II: A new challenge for reinforcement learning. *CoRR*, 2017. URL http://arxiv.org/abs/1708.04782.

Wang, H., Keskar, N. S., Xiong, C., and Socher, R. Identifying generalization properties in neural networks. 2018a. URL https://openreview.net/forum?id=BJxOHs0cKm.

Wang, J., Liu, Y., and Li, B. Reinforcement learning with perturbed rewards. *CoRR*, abs/1810.01032, 2018b. URL http://arxiv.org/abs/1810.01032.

Whiteson, S., Tanner, B., Taylor, M. E., and Stone, P. Protecting against evaluation overfitting in empirical reinforcement learning. *2011 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, 2011.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 1992.

Zhan, Y. and Taylor, M. E. Online transfer learning in reinforcement learning domains. *CoRR*, abs/1507.00436, 2015.

Zhang, A., Ballas, N., and Pineau, J. A dissection of overfitting and generalization in continuous reinforcement learning. *CoRR*, 2018a. URL https://arxiv.org/abs/1806.07937.

Zhang, C., Vinyals, O., Munos, R., and Bengio, S. A study on overfitting in deep reinforcement learning. *CoRR*, 2018b. URL http://arxiv.org/abs/1804.06893.