# A Survey on Generalization in Deep Reinforcement Learning

**Ezgi Korkmaz**
University College London
ezgikorkmazmail@gmail.com

## Abstract

Reinforcement learning research obtained significant success and attention with the utilization of deep neural networks to solve problems in high dimensional state or action spaces. While deep reinforcement learning policies are currently being deployed in many different fields from medical applications to self driving vehicles, there are still ongoing questions the field is trying to answer on the generalization capabilities of deep reinforcement learning policies. In this paper, we will go over the fundamental reasons why deep reinforcement learning policies encounter overfitting problems that limit their generalization capabilities. Furthermore, we will formalize and unify the diverse solution approaches to increase generalization, and overcome overfitting in deep reinforcement learning policies. We believe our study can provide a compact systematic unified analysis for the current advancements in deep reinforcement learning, and help to construct robust deep neural policies with improved generalization abilities.

## 1   Introduction

The performance of reinforcement learning algorithms has been boosted with the utilization of deep neural networks as function approximators (Mnih et al., 2015). Currently, it is possible to learn deep reinforcement learning policies that can operate in large state and/or action space MDPs. This progress consequently resulted in building reasonable deep reinforcement learning policies that can play computer games with high dimensional state representations (e.g. Atari, StarCraft), solve complex robotics control tasks, design algorithms (Mankowitz et al., 2023; Fawzi et al., 2022), and play some of the most complicated board games (e.g. Chess, Go) (Schrittwieser et al., 2020). However, deep reinforcement learning algorithms also experience several problems caused by their overall generalization capabilities. Some studies demonstrated these problems via adversarial perturbations introduced to the state observations of the policy (Huang et al., 2017; Kos & Song, 2017; Korkmaz, 2022), several focused on exploring the fundamental issues with function approximation, estimation biases in the state-action value function (Hasselt et al., 2016), or with new architectural design ideas (Wang et al., 2016).

The fact that we are not able to explore the whole MDP for high dimensional state representation MDPs, even with deep neural networks as function approximators, is one of the root problems that limits generalization. On top this, some portion of the problems are directly caused by the utilization of deep neural networks and thereby the intrinsic problems inherited from their utilization (Goodfellow et al., 2015; Szegedy et al., 2014).

In this paper we will focus on generalization in deep reinforcement learning and underlying causes of the limitations deep reinforcement learning research currently faces. In particular, we will try to answer the following questions:

- *What is the role of exploration in overfitting for deep reinforcement learning?*

- *What are the causes of overestimation bias observed in state-action value functions?*

- *What has been done to overcome the overfitting problems that deep reinforcement learning algorithms have encountered so far?*

- *What future directions are there for reinforcement learning research to obtain higher level generalization abilities for deep neural policies?*

To answer these questions we go through research from several subfields on overfitting and generalization in reinforcement learning. We introduce a categorization of the different methods used to both achieve and test generalization, and use it to systematically summarize and consolidate the current body of research. We further describe the issue of value function overestimation, and the role of exploration in overfitting in reinforcement learning. Furthermore, we explain new emerging research areas that can potentially target these questions in the long run including meta-reinforcement learning and lifelong learning. We hope that our paper can provide a compact overview and unification of the current advancements and limitations in the field.

## 2 Preliminaries on Deep Reinforcement Learning

The aim in deep reinforcement learning is to learn a policy via interacting with an environment in a Markov Decision Process (MDP) that maximize expected cumulative discounted rewards. An MDP is represented by a tuple $\mathcal{M} = (S, A, P, r, \rho_0, \gamma)$, where $S$ represents the state space, $A$ represents the action space, $r : S \times A \to \mathbb{R}$ is a reward function, $\mathcal{P} : S \times A \to \Delta(S)$ is a transition probability kernel, $\rho_0$ represents the initial state distribution, $\gamma$ represents the discount factor. The objective in reinforcement learning is to learn a policy $\pi : S \to \Delta(A)$ which maps states to probability distributions on actions in order to maximize the expected cumulative reward $R = \mathbb{E} \sum_{t=0}^{T-1} \gamma^t r(s_t, a_t)$ where $a_t \sim \pi(s_t), s_{t+1} \sim \mathcal{P}(s_t, a_t)$. In $Q$-learning the goal is to learn the optimal state-action value function (Watkins, 1989)

$$Q^*(s, a) = R(s, a) + \sum_{s' \in S} P(s'|s, a) \max_{a' \in A} Q^*(s', a'). \tag{1}$$

This is achieved via iterative Bellman update which updates $Q(s_t, a_t)$ by

$$Q(s_t, a_t) + \alpha[\mathcal{R}_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)].$$

Thus, the optimal policy is determined by choosing the action $a^*(s) = \arg\max_a Q(s, a)$ in state $s$. In high dimensional state space or action space MDPs the optimal policy is decided via a function-approximated state-action value function represented by a deep neural network. In a parallel line of algorithm families the policy itself is directly parametrized by $\pi_\theta$, and the gradient estimator used in learning is

$$g = \mathbb{E}_t \big[ \nabla_\theta \log \pi_\theta(s_t, a_t)(Q(s_t, a_t) - \max_a Q(s_t, a)) \big]$$

where $Q(s_t, a_t)$ refers to the state-action value function at timestep $t$.

## 3 How to Achieve Generalization?

To be able to categorize different paths to achieve generalization first we will provide a definition meant to capture the behavior of a generic reinforcement learning algorithm.

**Definition 3.1.** A reinforcement learning training algorithm $\mathcal{A}$ learns a policy $\pi$ by interacting with an MDP $\mathcal{M}$. We divide up the execution of $\mathcal{A}$ into discrete time steps as follows. At each time $t$, the algorithm chooses a state $s_t$, takes an action $a_t$, observes a transition to state $s'_t$ with corresponding reward $r_t = r(s_t, a_t, s'_t)$. We define the history of algorithm $\mathcal{A}$ in MDP $\mathcal{M}$ to be the sequence $H_t = (s_0, a_0, s'_0, r_0), \dots (s_t, a_t, s'_t, r_t)$ of all the transitions observed by the algorithm so far. We require that state and action $(s_t, a_t)$ chosen at time $t$ are a function only of $H_{t-1}$, i.e the transitions observed so far by $\mathcal{A}$. At time $t = T$, the algorithm stops and outputs a policy $\pi$.

---

[0]DeepMind Control Suite

Intuitively, a reinforcement learning algorithm performs a sequence of queries $(s_t, a_t)$ to the MDP, and observes the resulting state transitions and rewards. In order to be as generic as possible, the definition makes no assumptions about how the algorithm chooses the sequence of queries. Notably, if taking action $a_t$ in state $s_t$ leads to a transition to state $s'_t$, there is no requirement that $s_{t+1} = s'_t$. Indeed, the only assumption is that $(s_{t+1}, a_{t+1})$ depends only on $H_t$, the history of transitions observed so far. This allows the definition to capture deep reinforcement learning algorithms, which may choose to query states and actions in a complex way based on previously observed state transitions. Based on this definition of generic reinforcement learning algorithm, we will now further define the different techniques proposed to achieve generalization.

**Definition 3.2.** Let $\mathcal{A}$ be a training algorithm that takes as input an MDP and outputs a policy. Given an MDP $\mathcal{M} = (S, A, P, r, \rho_0, \gamma)$, a *rewards transforming* generalization method $\mathcal{G}_R$ is given by a sequence of functions $F_t : (S \times A \times S \times \mathbb{R})^t \to \mathbb{R}$. The method attempts to achieve generalization by running $\mathcal{A}$ on MDP $\mathcal{M}$, but modifying the rewards at each time $t$ to be $\tilde{r}_t(s_t, a_t, s'_t) = F_{t-1}(H_{t-1})$, where $H_{t-1}$ is the history of algorithm $\mathcal{A}$ when running with the perturbed rewards.

In summary, a rewards transforming generalization methods simply runs the original algorithm, but modifies the observed rewards. Similarly, we define two additional generalization methods which run the original algorithm while modifying states and transition probabilities respectively.

**Definition 3.3.** Let $\mathcal{A}$ be a training algorithm that takes as input an MDP and outputs a policy. Given an MDP $\mathcal{M} = (S, A, P, r, \rho_0, \gamma)$, a *state transforming* generalization method $\mathcal{G}_S$ is given by a sequence of functions $F_t : (S \times A \times S \times \mathbb{R})^t \times S \to S$. The method attempts to achieve generalization by running $\mathcal{A}$ on MDP $\mathcal{M}$, but modifying the state chosen at time $t$ to be $\tilde{s}_t = F_{t-1}(H_{t-1}, s_t)$, where $H_{t-1}$ is the history of algorithm $\mathcal{A}$ when running with the perturbed states.

**Definition 3.4.** Let $\mathcal{A}$ be a training algorithm that takes as input an MDP and outputs a policy. Given an MDP $\mathcal{M} = (S, A, P, r, \rho_0, \gamma)$, a *transition probability transforming* generalization method $\mathcal{G}_{\mathcal{P}}$ is given by a sequence of functions $F_t : (S \times A \times S \times \mathbb{R})^t \times (S \times A \times S) \to \mathbb{R}$. The method attempts to achieve generalization by running $\mathcal{A}$ on MDP $\mathcal{M}$, but modifying the transition probabilities at time $t$ to be $\tilde{P}(s_t, a_t, s'_t) = F_{t-1}(H_{t-1}, s_t, a_t, s'_t)$, where $H_{t-1}$ is the history of algorithm $\mathcal{A}$ when running with the perturbed transition probabilities.

The last type of generalization method we define is based on directly modifying the way in which the training algorithm chooses the state and action pair for the next time step. While this definition is broad enough to capture very complex changes to the training algorithm, in practice the choice of modification generally has a simple description.

**Definition 3.5.** Let $\mathcal{A}$ be a training algorithm that takes as input an MDP and outputs a policy. Given an MDP $\mathcal{M} = (S, A, P, r, \rho_0, \gamma)$, a *policy transforming* generalization method $\mathcal{G}_\pi$ is given by a sequence of functions $F_t : (S \times A \times S \times \mathbb{R})^t \to S \times A$. The method attempts to achieve generalization by running $\mathcal{A}$ on MDP $\mathcal{M}$, but modifying the policy by which $\mathcal{A}$ chooses the next state and action to be $(\tilde{s}_t, \tilde{a}_t) = F_{t-1}(H_{t-1})$, where $H_{t-1}$ is the history of algorithm $\mathcal{A}$ when running with the perturbed policy.

All the definitions so far categorize methods to modify training algorithms in order to achieve generalization. However, many such methods for modifying training algorithms have a corresponding method which can be used to test the generalization capabilities of a trained policy. Our final definition captures this correspondence.

**Definition 3.6.** Let $\hat{\pi}$ be a trained policy for an MDP $\mathcal{M}$. Let $F_t$ be a sequence of functions corresponding to a generalization method from one of the previous definitions. The *generalization testing* method of $F_t$ is given by executing the policy $\hat{\pi}$ in $\mathcal{M}$, but in each time step applying the modification $F_t$ where the history $H_t$ is given by the transitions executed by $\hat{\pi}$ so far. When both a generalization method and a generalization testing method are used concurrently, we will use subscripts to denote the generalization method and superscripts to denote the testing method. For instance, $\mathcal{G}_S^\pi$ corresponds to training with a state transforming method, and testing with a policy transforming method.

# 4 Roots of Overestimation in Deep Reinforcement Learning

Many reinforcement learning algorithms compute estimates for the state-action values in an MDP. Because these estimates are usually based on a stochastic interaction with the MDP, computing

Table 1: Environment and algorithm details for different exploration strategies for generalization.

| Citation | Proposed Method | Environment | Reinforcement Learning Algorithm |
|---|---|---|---|
| Bellemare et al. (2016) | Count-based | ALE | A3C and DQN |
| Osband et al. (2016b) | RLSVI | Tetris | Tabular $Q$-learning |
| Osband et al. (2016a) | Bootstrapped DQN | ALE | DQN |
| Houthooft et al. (2017) | VIME | DeepMind Control Suite | TRPO |
| Fortunato et al. (2018) | NoisyNet | ALE | A3C and DQN |
| Lee et al. (2021) | SUNRISE | DCS[1]and Atari | Soft Actor-Critic and Rainbow DQN |

accurate estimates that correctly generalize to further interactions is one of the most fundamental tasks in reinforcement learning. A major challenge in this area has been the tendency of many classes of reinforcement learning algorithms to consistently overestimate state-action values. Initially the overestimation bias for $Q$-learning is discussed and theoretically justified by Thrun & Schwartz (1993) as a biproduct of using function approximators for state-action value estimates. Following this initial discussion it has been shown that several parts of the deep reinforcement learning process can cause overestimation bias. Learning overestimated state-action values can be caused by statistical bias of utilizing a single max operator (van Hasselt, 2010), coupling between value function and the optimal policy (Raileanu & Fergus, 2021; Cobbe et al., 2021), or caused by the accumulated function approximation error (Boyan & Moore, 1994).

Several methods have been proposed to target overestimation bias for value iteration algorithms. In particular, to solve this overestimation bias introduced by the max operator (van Hasselt, 2010) proposed to utilize a double estimator for the state-action value estimates. Later, the authors also created a version of this algorithm that can solve high dimensional state space problems (Hasselt et al., 2016). Some of the work on this line of research targeting overestimation bias for value iteration algorithms is based on simply averaging the state-action values with previously learned state-action value estimates during training time (Anschel et al., 2017).

While overestimation bias was demonstrated to be a problem and discussed over a long period of time (Thrun & Schwartz, 1993; van Hasselt, 2010), recent studies also further demonstrated that actor critic algorithms also suffers from this issue (Fujimoto et al., 2018).

## 5 The Role of Exploration in Overfitting

The fundamental trade-off of exploration vs exploitation is the dilemma that the agent can try to take actions to move towards more unexplored states by sacrificing the current immediate rewards. While there is a significant body of studies on provably efficient exploration strategies the results from these studies do not necessarily directly transfer to the high dimensional state or action MDPs. The most prominent indication of this is that, even though it is possible to use deep neural networks as function approximators for large state spaces, the agent will simply not be able to explore the full state space. The fact that the agent is able to only explore a portion of the state space simply creates a bias in the learnt value function (Baird, 1995).

In this section, we will go through several exploration strategies in deep reinforcement learning and how they affect policy overfitting. A quite simple version of this is based on adding noise in action selection during training e.g. $\epsilon$-greedy exploration. Note that this is an example of a policy transforming generalization method $\mathcal{G}_\pi$ in Definition 3.5 in Section 3. Yet it has been proven that to explore the state space these algorithms may take exponentially long (Kakade, 2003). Several others focused on randomizing different components of the reinforcement learning training algorithms. In particular, (Osband et al., 2016b) proposes the randomized least squared value iteration algorithm to explore more efficiently in order to increase generalization in reinforcement learning for linearly parametrized value functions. This is achieved by simply adding Gaussian noise as a function of state visitation frequencies to the training dataset. Later, the authors also propose the bootstrapped DQN algorithm (i.e. adding temporally correlated noise) to increase generalization with non-linear function approximation (Osband et al., 2016a).

Houthooft et al. (2017) proposed an exploration technique centered around maximizing the information gain on the agent's belief of the environment dynamics. In practice, the authors use Bayesian neural networks for effectively exploring high dimensional action space MDPs. Following this line of work on increasing efficiency during exploration Fortunato et al. (2018) proposes to add parametric

Table 2: Environment and algorithm details for data augmentation techniques for state observation generalization. All of the studies in this section focuses on state transformation methods $\mathcal{G}_S$ defined in Section 3.

| Citation | Proposed Method | Environment | Reinforcement Learning Algorithm |
|---|---|---|---|
| Yarats et al. (2021) | DrQ | DeepMind Control Suite, ALE | DQN |
| Laskin et al. (2020b) | CuRL | DeepMind Control Suite, ALE | Soft Actor Critic and DQN |
| Laskin et al. (2020a) | RAD | DeepMind Control Suite, ProcGen | Soft Actor Critic and PPO |
| Wang et al. (2020) | Mixreg | ProcGen | DQN and PPO |

noise to the deep reinforcement learning policy weights in high dimensional state MDPs. While several methods focused on ensemble state-action value function learning (Osband et al., 2016a), Lee et al. (2021) proposed reweighting target Q-values from an ensemble of policies (i.e. weighted Bellman backups) combined with highest upper-confidence bound action selection. Another line of research in exploration strategies focused on *count-based methods* that use the direct count of state visitations. In this line of work, Bellemare et al. (2016) tried to lay out the relationship between count based methods and intrinsic motivation, and used count-based methods for high dimensional state MDPs (i.e. Arcade Learning Environment). Yet it is worthwhile to note that most of the current deep reinforcement learning algorithms use very simple exploration techniques such as $\epsilon$-greedy which is based on taking the action maximizing the state-action value function with probability $1 - \epsilon$ and taking a random action with probability $\epsilon$ (Mnih et al., 2015; Hasselt et al., 2016; Wang et al., 2016; Hamrick et al., 2020; Kapturowski et al., 2023).

It is possible to argue that the fact that the deep reinforcement learning policy obtained higher score with the same number of samples by a particular type of training method $\mathcal{A}$ compared to method $\mathcal{B}$ is by itself evidence that the technique $\mathcal{A}$ leads to more generalized policies. Even though the agent is trained and tested in the same environment, the explored states during training time are not exactly the same states visited during test time. The fact that the policy trained with technique $\mathcal{A}$ obtains a higher score at the end of episode is sole evidence that the agent trained with $\mathcal{A}$ was able to visit further states in the MDP and thus succeed in them. Yet, throughout the paper we will discuss different notions of generalization investigated in different subfields of reinforcement learning research. While exploration vs exploitation stands out as one of the main problems in reinforcement learning policy performance most of the work conducted in this section still is not able to obtain policies that perform as well as those in the studies described in Section 6.

# 6 Regularization

In this section we will focus on different regularization techniques employed to increase generalization in deep reinforcement learning policies. We will go through these works by categorizing under data augmentation, adversarial training, and direct function regularization. Under each category we will connect these different line of approaches to increase generalization in deep reinforcement learning to the settings we defined in Section 3.

## 6.1 Data Augmentation

Several studies focus on diversifying the observations of the deep reinforcement learning policy to increase generalization capabilities. A line of research in this regard focused on simply employing versions of data augmentation techniques (Laskin et al., 2020a,b; Yarats et al., 2021) for high dimensional state representation environments. In particular, these studies involve simple techniques such as cropping, rotating or shifting the state observations during training time. While this line of work got considerable attention, a quite recent study (Agarwal et al., 2021b) demonstrated that when the number of random seeds is increased to one hundred the relative performance achieved and reported in the original papers of (Laskin et al., 2020b; Yarats et al., 2021) on data augmentation training in deep reinforcement learning decreases to a level that is significant to mention.

While some of the work on this line of research simply focuses on using a set of data augmentation methods (Laskin et al., 2020a,b; Yarats et al., 2021), other work focuses on proposing new environments to train in (Cobbe et al., 2020). The studies on designing new environments to train deep reinforcement learning policies basically aim to provide high variation in the observed environment

Table 3: Environment and algorithm details for different direct function regularization strategies for trying to overcome overfitting problems in reinforcement learning. Note that most of the methods based on direct function regularization are a form of policy perturbation method $\mathcal{G}_\pi$ to overcome overfitting as described in Section 3.

| Citation | Proposed Method | Environment | Reinforcement Learning Algorithm |
|---|---|---|---|
| Igl et al. (2019) | SNI and IBAC | GridWorld and CoinRun | Proximal Policy Optimization |
| Vieillard et al. (2020b) | Munchausen RL | Atari | DQN and IQN |
| Lee et al. (2020) | Network Randomization | 2D CoinRun and 3D DeepMind Lab | Proximal Policy Optimization |
| Amit et al. (2020) | Discount Regularization | GridWorld and Mujoco[2] | Twin Delayed DDPG (TD3) |
| Agarwal et al. (2021a) | PSM | DDMC and Rectangle Game[3] | DrQ |
| Liu et al. (2021) | BN and dropout and $L_2/L_1$ | Mujoco | PPO, TRPO, SAC, A2C |

such as changing background colors and changing object shapes in ways that are meaningful in the game, in order to increase test time generalization.

Within this category some work focuses on producing more observations by simply blending in (e.g. creating a mixture state from multiple different observations) several observations to increase generalization (Wang et al., 2020). While most of the studies trying to increase generalization by data augmentation techniques are primarily conducted in the DeepMind Control Suite or the Arcade Learning Environment (ALE) (Bellemare et al., 2013), some small fraction of these studies (Wang et al., 2020) are conducted in relatively recently designed training environments like ProcGen (Cobbe et al., 2020). Cobbe et al. (2019) focuses on decoupling the training and testing set for reinforcement learning via simply proposing a new game environment CoinRun.

## 6.2 Direct Function Regularization

While some of the work we have discussed so far focuses on regularizing the data (i.e. state observations) as in Section 6.1, some focuses on directly regularizing the function learned with the intention of simulating techniques from deep neural network regularization like batch normalization and dropout (Igl et al., 2019). While some studies have attempted to simulate these known techniques in reinforcement learning, some focus on directly applying them to overcome overfitting. In this line of research, Liu et al. (2021) proposes to use known techniques from deep neural network regularization to apply in continous control deep reinforcement learning training. In particular, these techniques are batch normalization (BN) (Ioffe & Szegedy, 2015), weight clipping, dropout, entropy and $L_2/L_1$ weight regularization.

Lee et al. (2020) proposes to utilize a random network to randomize the input observations to increase generalization skills of deep reinforcement learning policies, and tests the proposal in 2D CoinRun game proposed by Cobbe et al. (2019) and 3D DeepMind Lab. In particular, the authors basically introduce a random convolutional layer to perturb the state observations. Hence, this study is also a clear example of a state transformation generalization method $\mathcal{G}_S$ described in Definition 3.3. While this is another example of random state perturbation methods we will further explain in Section 6.3 the worst-case perturbation methods to target generalization in reinforcement learning policies.

Some work employs contrastive representation learning to learn deep reinforcement learning policies from state observations that are close to each other (Agarwal et al., 2021a). The authors of this study leverage the temporal aspect of reinforcement learning and propose a policy similarity metric. The main goal of the paper is to lay out the sequential structure and utilize representation learning to learn generalizable abstractions from state representations. One drawback of this study is that most of the experimental study is conducted in a non-baseline environment (Rectangle game). Even though the authors show surprising results for this particular game, it is not clear how the proposed method would work for high dimensional state representation MDPs such as the Arcade Learning Environment. Malik et al. (2021) studies query complexity of reinforcement learning policies that can generalize to multiple environments. The authors of this study focus on an example of the transition probability transformation setting $\mathcal{G}_\mathcal{P}$ in Definition 3.4, and the reward function transformation setting $\mathcal{G}_R$ in Definition 3.2.

---

[2]Low dimensional setting of Mujoco is used for this study.

[3]Rectangle game is a simple video game with only two actions, "Right" and "Jump". The game has black background and two rectangles where the goal of the game is to avoid white obstacles and reach to the right side of the screen. Agarwal et al. (2021a) is the only paper we encountered experimenting with this particular game.

Table 4: Environment and algorithm details for adversarial policy regularization and attack techniques in deep reinforcement learning. Note that most of the methods based on adversarial policy regularization are a form of state observation perturbation method $\mathcal{G}_S^S$ as described in Definition 3.6.

| Citation | Proposed Method | Environment | Reinforcement Learning Algorithm |
|---|---|---|---|
| Huang et al. (2017) | FGSM | ALE | DQN, TRPO, A3C |
| Kos & Song (2017) | FGSM | ALE | DQN and IQN |
| Lin et al. (2017) | Strategically-Timed Attack | ALE | A3C and DQN |
| Gleave et al. (2020) | Adversarial Policies | Mujoco | Proximal Policy Optimization |
| Huan et al. (2020) | SA-DQN | ALE and $L_{\text{Mujoco}}$[4] | DDQN and PPO |
| Korkmaz (2021d) | KMAP and HMAP | ALE | DDQN |
| Korkmaz (2022) | Adversarial Framework | ALE | DDQN and A3C |
| Korkmaz (2023) | Natural Attacks | ALE | DDQN and A3C |
| Korkmaz & Brown-Cohen (2023) | Adversarial Detection | ALE | DDQN |

Another line of study in direct function generalization investigates the relationship between reduced discount factor and adding an $\ell_2$-regularization term to the loss function (weight decay) (Amit et al., 2020). The authors in this work demonstrate the explicit connection between reducing the discount factor and adding an $\ell_2$-regularizer to the value function for temporal difference learning. In particular, this study demonstrates that adding an $\ell_2$-regularization term to the loss function is equal to training with a lower discount term, which the authors refer to as *discount regularization*. The results of this study however are based on experiments from tabular reinforcement learning, and the low dimensional setting of the Mujoco environment.

On the reward transformation for generalization setting $\mathcal{G}_R$ defined in Definition 3.2, Vieillard et al. (2020b) adds the scaled log policy to the current rewards. To overcome overfitting some work tries to learn explicit or implicit similarity between the states to obtain a reasonable policy Lan et al. (2021). In particular, the authors in this work try to unify the state space representations by providing a taxonomy of metrics in reinforcement learning. Several studies proposed different ways to include Kullback-Leibler (KL) divergence between the current policy and the pre-updated policy to add as a regularization term in the reinforcement learning objective (Schulman et al., 2015). Recently, some studies argued that utilizing Kullback-Leibler regularization implicitly averages the state-action value estimates (Vieillard et al., 2020a).

## 6.3 The Adversarial Perspective for Deep Neural Policy Generalization

One of the ways to regularize the state observations is based on considering worst-case perturbations added to state observations (i.e. adversarial perturbations). This line of work starts with introducing perturbations produced by the fast gradient sign method proposed by Goodfellow et al. (2015) into deep reinforcement learning observations at test time Huang et al. (2017) Kos & Song (2017), and compares the generalization capabilities of the trained deep reinforcement learning policies in the presence worst-case perturbations and Gaussian noise. These gradient based adversarial methods are based on taking the gradient of the cost function used to train the policy with respect to the state observation. Several other techniques have been proposed on the optimization line of the adversarial alteration of state observations. In this line of work, Korkmaz (2020) suggested a Nesterov momentum-based method to produce adversarial perturbations for deep reinforcement learning policies. Korkmaz (2022) showed that deep reinforcement learning policies learn shared adversarial features across MDPs. In this work the authors investigate the root causes of this problem, and demonstrate that policy high-sensitivity directions and the perceptual similarity of the state observations are uncorrelated. Furthermore, the study demonstrates that the current state-of-the-art adversarial training techniques also learn similar high-sensitivity directions as the vanilla trained deep reinforcement learning policies.[5] While some studies focused on state observation alterations to assess policy resilience with respect to these changes, some studies focused on interpretability and explainability of these changes in these state observation alterations and how these alterations

---

[4]Low dimensional state Mujoco refers to the setting of Mujoco where the state dimensions are not represented by pixels and dimensions of the state observations range from 11 to 117.

[5]From the security point of view, this adversarial framework is under the category of black-box adversarial attacks in which this is the first study that demonstrated that the deep reinforcement learning policies are vulnerable to black-box adversarial attacks (Korkmaz, 2022). Furthermore, note that black-box adversarial perturbations are more generalizable global perturbations that can effect many different policies.

have different effects on standard deep reinforcement learning training algorithms and certified (i.e. adversarial) training algorithms Korkmaz (2021d)[6]. Note that this line of work falls under the state observation generalization testing category $\mathcal{G}_S^S$ provided in Definition 3.6.

While several studies focused on improving optimization techniques to compute optimal perturbations, a line of research focused on making deep neural policies resilient to these perturbations. Pinto et al. (2017) proposed to model the dynamics between the adversary and the deep neural policy as a zero-sum game where the goal of the adversary is to minimize expected cumulative rewards of the deep reinforcement learning policy. This study is a clear example of transition probability perturbation to achieve generalization $\mathcal{G}_{\mathcal{P}}$ in Definition 3.4 of Section 3. Gleave et al. (2020) approached this problem with an adversary model which is restricted to take natural actions in the MDP instead of modifying the observations with $\ell_p$-norm bounded perturbations. The authors model this dynamic as a zero-sum Markov game and solve it via self play Proximal Policy Optimization (PPO). Some recent studies, proposed to model the interaction between the adversary and the deep reinforcement learning policy as a state-adversarial MDP, and claimed that their proposed algorithm State Adversarial Double Deep Q-Network (SA-DDQN) learns theoretically certified robust policies against natural noise and perturbations. In particular, these certified adversarial training techniques aim to add a regularizer term to the temporal difference loss in deep $Q$-learning.

$$\mathcal{H}(r_i + \gamma \max_a \hat{Q}_{\hat{\theta}}(s_i, a; \theta) - Q_\theta(s_i, a_i; \theta)) + \kappa \mathcal{R}(\theta)$$

where $\mathcal{H}$ is the Huber loss, $\hat{Q}$ refers to the target network and $\kappa$ is to adjust the level of regularization for convergence. The regularizer term can vary for different certified adversarial training techniques yet the baseline technique uses

$$\mathcal{R}(\theta) = \max\{\max_{\hat{s} \in B(s)} \max_{a \neq \arg\max_{a'} Q(s,a')} Q_\theta(\hat{s}, a) - Q_\theta(\hat{s}, \arg\max_{a'} Q(s, a')), -c\}.$$

where $B(s)$ is an $\ell_p$-norm ball of radius $\epsilon$. While these certified adversarial training techniques drew some attention from the community, more recently manifold concerns have been raised on the robustness of theoretically certified adversarially trained deep reinforcement learning policies (Korkmaz, 2021c,d). In these studies, the authors argue that adversarially trained (i.e. certified robust) deep reinforcement learning policies learn inaccurate state-action value functions and non-robust features from the environment. In particular, in Korkmaz (2021c) the authors use action manipulation to investigate worst-case perturbation training. This study is also a clear example of a policy perturbation generalization testing method $\mathcal{G}_S^\pi$ in Definition 3.6. More importantly, recently it has been shown that adversarially trained deep reinforcement learning policies have worse generalization capabilities compared to vanilla trained reinforcement learning policies in high dimensional state space MDPs (Korkmaz, 2023)[7]. While this study provides a contradistinction between adversarial directions and natural directions that are intrinsic to the MDP, it further demonstrates that the certified adversarial training techniques block generalization capabilities of standard deep reinforcement learning policies. Furthermore note that this study is also a clear example of a state observation perturbation generalization testing method $\mathcal{G}_S^S$ the Definition 3.6 in Section 3.

## 7   Meta-Reinforcement Learning and Meta Gradients

A quite recent line of research directs its research efforts to discovering reinforcement learning algorithms automatically, without explicitly designing them, via meta-gradients (Oh et al., 2020; Xu et al., 2020). This line of study targets learning the "learning algorithm" by only interacting with a set of environments as a meta-learning problem. In particular,

$$\eta^* = \arg\max_\eta \mathbb{E}_{\varepsilon \sim \rho(\varepsilon)} \mathbb{E}_{\theta_0 \sim \rho(\theta_0)} [\mathbb{E}_{\theta_N} [\sum_{t=0}^{\infty} \gamma^t r_t]]$$

here the optimal update rule is parametrized by $\eta$, for a distribution on environments $\rho(\varepsilon)$ and initial policy parameters $\rho(\theta_0)$ where $\mathbb{E}_{\theta_N} [\sum_{t=0}^{\infty} \gamma^t r_t]$ is the expected return for the end of the lifetime of the agent.

---

[6]See an initial and preliminary version of the paper Korkmaz (2021d) here Korkmaz (2021a)

[7]A short and preliminary version of the paper (Korkmaz, 2023) can also be found here (Korkmaz, 2021b)

The overarching objective of meta-reinforcement learning is to be able to build agents that can learn *how to learn* over time, thus allowing these policies to adapt to a changing environment or even any other changing conditions of the MDP. Quite recently, a significant line of research has been conducted to achieve this objective, particularly Oh et al. (2020) proposes to discover the value function in reinforcement learning. Following this work Xu et al. (2020) proposed a joint meta-learning framework to learn what the policy should predict and how these predictions should be used in updating the policy. Recently, Kirsch et al. (2022) proposes to use symmetry information in discovering reinforcement learning algorithms and discusses meta-generalization. There is also some work on enabling reinforcement learning algorithms to discover temporal abstractions (Veeriah et al., 2021). In particular, temporal abstraction refers to the ability of the policy to abstract a sequence of actions to achieve certain sub-tasks. As it stands now meta-reinforcement learning can be considered to be a promising research direction that could enable us to build deep reinforcement learning policies that can generalize to different environments, to changing environments over time, or even to different tasks.

## 8 Transfer in Reinforcement Learning

Transfer in reinforcement learning is a subfield heavily discussed in certain applications of reinforcement learning algorithms e.g. robotics. In current robotics research there is not a safe way of training a reinforcement learning agent by letting the robot explore in real life. Hence, the way to overcome this to train policies in a simulated environment, and install the trained policies in the actual application setting. The fact that the simulation environment and the installation environment are not identical is one of the main problems for reinforcement learning application research. This is referred to as the *sim-to-real gap*.

Another subfield in reinforcement learning research focusing on obtaining generalizable policies investigates this concept through *transfer in reinforcement learning*. The consideration in this line of research is to build policies that are trained for a particular task with limited data and to try to make these policies perform well on slightly different tasks. An initial discussion on this starts with (Taylor & Stone, 2007) to obtain policies initially trained in a source task and transferred to a target task in a more sample efficient way. Later, Tirinzoni et al. (2018) proposes to transfer value functions that are based on learning a prior distribution over optimal value functions from a source task. However, this study is conducted in simple environments with low dimensional state spaces. (Barreto et al., 2017) considers the reward transformation setting $\mathcal{G}_R$ in Definition 3.2 from Section 3. In particular, the authors consider a policy transfer between a specific task with a reward function $r(s, a)$ and a different task with reward function $r'(s, a)$. The goal of the study is to decouple the state representations from the task. In the setting of state transformation for generalization $\mathcal{G}_S$ in Definition 3.3 Gamrian & Goldberg (2019) focuses on state-wise differences between source and target task. In particular, the authors use unaligned generative adversarial networks to create target task states from source task states. In the setting of policy transformation for generalization $\mathcal{G}_\pi$ in Definition 3.5 Jain et al. (2020) focuses on zero-shot generalization to a newly introduced action set to increase adaptability.

While transfer learning is a promising research direction for reinforcement learning, the studies in this subfield still remain oriented only towards reinforcement learning applications, and it is possible to consider the research centered on this subfield as not at the same level of maturity as the previously discussed line of research in Section 6 in terms of being able to test the claims or propositions in complex established baselines.

## 9 Lifelong Reinforcement Learning

*Lifelong learning* is a subfield closely related to transfer learning that has recently drawn attention from the reinforcement learning community. Lifelong learning aims to build policies that can sequentially solve different tasks by being able to transfer knowledge between tasks. On this line of research, Lecarpentier et al. (2021) provide an algorithm for value-based transfer in the Lipschitz continuous task space with theoretical contributions for lifelong learning goals. In the setting of action transformation for generalization $\mathcal{G}_\pi$ in Definition 3.5 Chandak et al. (2020) focuses on temporally varying (e.g. variations between source task and target task) the action set in lifelong learning. In lifelong reinforcement learning some studies focus on different exploration strategies. In particular,

Garcia & Thomas (2019) models the exploration strategy problem for lifelong learning as another MDP, and the study uses a separate reinforcement learning agent to find an optimal exploration method for the initial lifelong learning agent. The lack of benchmarks limits the progress of lifelong reinforcement learning research by restricting the direct comparison between proposed algorithms or methods. However, quite recent work proposed a new training environment benchmark based on robotics applications for lifelong learning to overcome this issue (Wolczyk et al., 2021)[8].

## 10  Conclusion

In this paper we tried to answer the following questions: *(i) What are the explicit problems limiting reinforcement learning algorithms from obtaining high-performing policies that can generalize? (ii) How can we categorize the different techniques proposed so far to achieve generalization in reinforcement learning? (iii) What are the similarities and differences of these different techniques proposed by different subfields of reinforcement learning research to build reinforcement learning policies that generalize?* To answer these questions first we explain the importance of exploration strategies in overfitting, and explain the manifold causes of overestimation bias in reinforcement learning. In the second part of the paper we propose a framework to unify and categorize the various techniques to achieve generalization in reinforcement learning. Starting from explaining all the different regularization techniques in either state representations or in learnt value functions from worst-case to average-case, we provide a current layout of the wide range of reinforcement learning subfields that are essentially working towards the same objective, i.e. generalizable deep reinforcement learning policies. Finally, we provided a discussion for each category on the drawbacks and advantages of these algorithms. We believe our study can provide a compact unifying formalization on recent reinforcement learning generalization research.

## References

Rishabh Agarwal, Marlos C. Machado, Pablo Samuel Castro, and Marc G. Bellemare. Contrastive behavioral similarity embeddings for generalization in reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2021a.

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C. Courville, and Marc G. Bellemare. Deep reinforcement learning at the edge of the statistical precipice. *Conference on Neural Information Processing Systems (NeurIPS)*, 2021b.

Ron Amit, Ron Meir, and Kamil Ciosek. Discount factor as a regularizer in RL. In *International Conference on Machine Learning (ICML)*, 2020.

Oron Anschel, Nir Baram, and Nahum Shimkin. Averaged-dqn: Variance reduction and stabilization for deep reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2017.

Leemon Baird. Residual algorithms: RL with function approximation. In *International Conference on Machine Learning (ICML)*, 1995.

André Barreto, Will Dabney, Rémi Munos, Jonathan J. Hunt, Tom Schaul, David Silver, and Hado van Hasselt. Successor features for transfer in reinforcement learning. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2017.

Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael. Bowling. The arcade learning environment: An evaluation platform for general agents. *JAIR*, 2013.

Marc G. Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Rémi Munos. Unifying count-based exploration and intrinsic motivation. *Conference on Neural Information Processing Systems (NeurIPS)*, 2016.

Justin A. Boyan and Andrew W. Moore. Generalization in rl: Safely approximating the value function. In *Conference on Neural Information Processing Systems (NeurIPS)*, 1994.

Yash Chandak, Georgios Theocharous, Chris Nota, and Philip S. Thomas. Lifelong learning with a changing action set. In *AAAI Conference on Artificial Intelligence, AAAI* , 2020.

---

[8]The state dimension for this benchmark is 12. Hence, the state space is quite low dimensional.

Karl Cobbe, Oleg Klimov, Christopher Hesse, Taehoon Kim, and John Schulman. Quantifying generalization in reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2019.

Karl Cobbe, Christopher Hesse, Jacob Hilton, and John Schulman. Leveraging procedural generation to benchmark reinforcement learning. *International Conference on MAchine Learning (ICML)*, 2020.

Karl Cobbe, Jacob Hilton, Oleg Klimov, and John Schulman. Phasic policy gradient. In *International Conference on Machine Learning (ICML)*, 2021.

Alhussein Fawzi, Matej Balog, Aja Huang, Thomas Hubert, Bernardino Romera-Paredes, Moham-madamin Barekatain, Alexander Novikov, Francisco J. R. Ruiz, Julian Schrittwieser, Grzegorz Swirszcz, David Silver, Demis Hassabis, and Pushmeet Kohli. Discovering faster matrix multipli-cation algorithms with reinforcement learning. *Nature*, 610(7930):47–53, 2022.

Meire Fortunato, Mohammad Gheshlaghi Azar, Bilal Piot, Jacob Menick, Ian Osband, Alex Graves, Vlad Mnih, Rémi Munos, Demis Hassabis, Olivier Pietquin, Charles Blundell, and Shane Legg. Noisy networks for exploration. *International Conference on Learning Representations (ICLR)*, 2018.

Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning (ICML)*, 2018.

Shani Gamrian and Yoav Goldberg. Transfer learning for related RL tasks via image-to-image translation. In *International Conference on Machine Learning (ICML)*, 2019.

Francisco M. Garcia and Philip S. Thomas. A meta-mdp approach to exploration for lifelong reinforcement learning. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

Adam Gleave, Michael Dennis, Cody Wild, Kant Neel, Sergey Levine, and Stuart Russell. Adversarial policies: Attacking deep RL. *International Conference on Learning Representations (ICLR)*, 2020.

Ian Goodfellow, Jonathan Shelens, and Christian Szegedy. Explaning and harnessing adversarial examples. *International Conference on Learning Representations (ICLR)*, 2015.

Jessica Hamrick, Victor Bapst, Alvaro SanchezGonzalez, Tobias Pfaff, Theophane Weber, Lars Buesing, and Peter Battaglia. Combining q-learning and search with amortized value estimates. In *8th International Conference on Learning Representations, ICLR*, 2020.

Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. *AAAI Conference on Artificial Intelligence, AAAI*, 2016.

Rein Houthooft, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. VIME: variational information maximizing exploration. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2017.

Zhang Huan, Chen Hongge, Xiao Chaowei, Bo Li, Mingyan Boning, Duane Liu, and ChoJui Hsiesh. Robust deep reinforcement learning against adversarial perturbations on state observatons. *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

Sandy Huang, Nicholas Papernot, Yan Goodfellow, Ian an Duan, and Pieter Abbeel. Adversarial attacks on neural network policies. *International Conference on Learning Representations (ICLR)*, 2017.

Maximilian Igl, Kamil Ciosek, Yingzhen Li, Sebastian Tschiatschek, Cheng Zhang, Sam Devlin, and Katja Hofmann. Generalization in reinforcement learning with selective noise injection and information bottleneck. *Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, 2015.

Ayush Jain, Andrew Szot, and Joseph J. Lim. Generalization to new actions in RL. In *International Conference on Machine Learning (ICML)*, 2020.

Sham Kakade. On the sample complexity of reinforcement learning. In *PhD Thesis*, 2003.

Steven Kapturowski, Victor Campos, Ray Jiang, Nemanja Rakicevic, Hado van Hasselt, Charles Blundell, and Adrià Puigdomènech Badia. Human-level atari 200x faster. In *The Eleventh International Conference on Learning Representations, ICLR 2023*, 2023.

Louis Kirsch, Sebastian Flennerhag, Hado van Hasselt, Abram L. Friesen, Junhyuk Oh, and Yutian Chen. Introducing symmetries to black box meta reinforcement learning. In *AAAI Conference on Artificial Intelligence, AAAI*, 2022.

Ezgi Korkmaz. Nesterov momentum adversarial perturbations in the deep reinforcement learning domain. *International Conference on Machine Learning (ICML) Workshop.*, 2020.

Ezgi Korkmaz. Adversarially trained neural policies in fourier domain. *International Conference on Learning Representation (ICLR) Robust and Reliable Machine Learning in the Real World Workshop*, 2021a.

Ezgi Korkmaz. Adversarial training blocks generalization in neural policies. *International Conference on Learning Representation (ICLR) Robust and Reliable Machine Learning in the Real World Workshop*, 2021b.

Ezgi Korkmaz. Inaccuracy of state-action value function for non-optimal actions in adversarially trained deep neural policies. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2021c.

Ezgi Korkmaz. Investigating vulnerabilities of deep neural policies. *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2021d.

Ezgi Korkmaz. Deep reinforcement learning policies learn shared adversarial features across mdps. *AAAI Conference on Artificial Intelligence, AAAI*, 2022.

Ezgi Korkmaz. Adversarial robust deep reinforcement learning requires redefining robustness. *AAAI Conference on Artificial Intelligence, AAAI*, 2023.

Ezgi Korkmaz and Jonah Brown-Cohen. Detecting adversarial directions in deep reinforcement learning to make robust decisions. In *International Conference on Machine Learning, ICML 2023*, volume 202 of *Proceedings of Machine Learning Research*, pp. 17534–17543. PMLR, 2023.

Jernej Kos and Dawn Song. Delving into adversarial attacks on deep policies. *International Conference on Learning Representations (ICLR)*, 2017.

Charline Le Lan, Marc G. Bellemare, and Pablo Samuel Castro. Metrics and continuity in reinforcement learning. In *AAAI Conference on Artificial Intelligence, AAAI*, 2021.

Michael Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel, and Aravind Srinivas. Rl with augmented data. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020a.

Michael Laskin, Aravind Srinivas, and Pieter Abbeel. CURL: contrastive unsupervised representations for reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2020b.

Erwan Lecarpentier, David Abel, Kavosh Asadi, Yuu Jinnai, Emmanuel Rachelson, and Michael L. Littman. Lipschitz lifelong RL. *AAAI Conference on Artificial Intelligence, AAAI*, 2021.

Kimin Lee, Kibok Lee, Jinwoo Shin, and Honglak Lee. Network randomization: A simple technique for generalization in deep reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2020.

Kimin Lee, Michael Laskin, Aravind Srinivas, and Pieter Abbeel. SUNRISE: A simple unified framework for ensemble learning in deep reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2021.

Yen-Chen Lin, Hong Zhang-Wei, Yuan-Hong Liao, Meng-Li Shih, ing-Yu Liu, and Min Sun. Tactics of adversarial attack on DRL agents. *IJCAI*, 2017.

Zhuang Liu, Xuanlin Li, and Trevor Darrell. Regularization matters in policy optimization - an empirical study on continuous control. In *International Conference on Learning Representations (ICLR)*, 2021.

Dhruv Malik, Yuanzhi Li, and Pradeep Ravikumar. When is generalizable reinforcement learning tractable? In *Conference on Neural Information Processing Systems (NeurIPS)*, 2021.

Daniel J. Mankowitz, Andrea Michi, Anton Zhernov, Marco Gelmi, Marco Selvi, Cosmin Paduraru, Edouard Leurent, Shariq Iqbal, Jean-Baptiste Lespiau, Alex Ahern, Thomas Köppe, Kevin Millikin, Stephen Gaffney, Sophie Elster, Jackson Broshear, Chris Gamble, Kieran Milan, Robert Tung, Minjae Hwang, Taylan Cemgil, Mohammadamin Barekatain, Yujia Li, Amol Mandhane, Thomas Hubert, Julian Schrittwieser, Demis Hassabis, Pushmeet Kohli, Martin A. Riedmiller, Oriol Vinyals, and David Silver. Faster sorting algorithms discovered using deep reinforcement learning. *Nature*, 618(7964):257–263, 2023.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, arc G Bellemare, Alex Graves, Martin Riedmiller, Andreas Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518: 529–533, 2015.

Junhyuk Oh, Matteo Hessel, Wojciech M. Czarnecki, Zhongwen Xu, Hado van Hasselt, Satinder Singh, and David Silver. Discovering reinforcement learning algorithms. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped DQN. *Conference on Neural Information Processing Systems (NeurIPS)*, 2016a.

Ian Osband, Benjamin Van Roy, and Zheng Wen. Generalization and exploration via randomized value functions. In *International Conference on Machine Learning (ICML)*, 2016b.

Lerrel Pinto, James Davidson, Rahul Sukthankar, and Abhinav Gupta. Robust adversarial reinforcement learning. *International Conference on Machine Learning (ICML)*, 2017.

Roberta Raileanu and Rob Fergus. Decoupling value and policy for generalization in reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2021.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy P. Lillicrap, and David Silver. Mastering atari, go, chess and shogi by planning with a learned model. *Nat.*, 588 (7839):604–609, 2020.

John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. Trust region policy optimization. *CoRR*, 2015.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dimutru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *International Conference on Learning Representations (ICLR)*, 2014.

Matthew E. Taylor and Peter Stone. Cross-domain transfer for RL. In *International Conference on Machine Learning (ICML)*, 2007.

Sebastian Thrun and Anton Schwartz. Issues in using function approximation for reinforcement learning. *In Fourth Connectionist Models Summer School*, 1993.

Andrea Tirinzoni, Rafael Rodriguez Sanchez, and Marcello Restelli. Transfer of value functions via variational methods. *Conference on Neural Information Processing Systems (NeurIPS)*, 2018.

Hado van Hasselt. Double q-learning. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2010.

Vivek Veeriah, Tom Zahavy, Matteo Hessel, Zhongwen Xu, Junhyuk Oh, Iurii Kemaev, Hado van Hasselt, David Silver, and Satinder Singh. Discovery of options via meta-learned subgoals. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2021.

Nino Vieillard, Tadashi Kozuno, Olivier Pietquin, Rémi Munos, and Matthieu Geist. Leverage the average: an analysis of KL regularization in reinforcement learning. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020a.

Nino Vieillard, Olivier Pietquin, and Matthieu Geist. Munchausen RL. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020b.

Kaixin Wang, Bingyi Kang, Jie Shao, and Jiashi Feng. Improving generalization in RL with mixture regularization. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Van Hasselt, Marc Lanctot, and Nando. De Freitas. Dueling network architectures for deep reinforcement learning. *International Conference on Machine Learning (ICML)*, pp. 1995–2003, 2016.

Chris Watkins. Learning from delayed rewards. In *PhD thesis, Cambridge*, 1989.

Maciej Wolczyk, Michal Zajac, Razvan Pascanu, Lukasz Kucinski, and Piotr Milos. Continual world: A robotic benchmark for continual reinforcement learning. *Conference on Neural Information Processing Systems (NeurIPS)*, 2021.

Zhongwen Xu, Hado Philip van Hasselt, Matteo Hessel, Junhyuk Oh, Satinder Singh, and David Silver. Meta-gradient reinforcement learning with an objective discovered online. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

Denis Yarats, Ilya Kostrikov, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International Conference on Learning Representations (ICLR)*, 2021.