A Survey of Exploration Methods in Reinforcement Learning

Susan Amin   SUSAN.AMIN@MAIL.MCGILL.CA
Department of Computer Science, McGill University
Mila- Québec Artificial Intelligence Institute
Montréal, Québec, Canada

Maziar Gomrokchi *   GOMROKMA@MILA.QUEBEC
Department of Computer Science, McGill University
Mila- Québec Artificial Intelligence Institute
Montréal, Québec, Canada

Harsh Satija *   HARSH.SATIJA@MAIL.MCGILL.CA
Department of Computer Science, McGill University
Mila- Québec Artificial Intelligence Institute
Montréal, Québec, Canada

Herke van Hoof *   H.C.VANHOOF@UVA.NL
Informatics Institute, University of Amsterdam
Amsterdam, the Netherlands

Doina Precup   DPRECUP@CS.MCGILL.CA
Department of Computer Science, McGill University
Mila- Québec Artificial Intelligence Institute
Montréal, Québec, Canada

Abstract

Exploration is an essential component of reinforcement learning algorithms, where agents need to learn how to predict and control unknown and often stochastic environments. Reinforcement learning agents depend crucially on exploration to obtain informative data for the learning process as the lack of enough information could hinder effective learning. In this article, we provide a survey of modern exploration methods in (Sequential) reinforcement learning, as well as a taxonomy of exploration methods.

Keywords: Exploration, Reinforcement Learning, Exploration-Exploitation Trade-off, Markov Decision Processes, Sequential Decision Making

1. Introduction

When a reinforcement learning (RL) agent starts acting in an environment, it usually does not have any prior knowledge regarding the task which it needs to tackle. The agent must interact with the environment, by taking actions and observing their consequences (in the

---

*. These authors contributed equally to the work

form of rewards and next states), and then it can use this data to improve its behavior, as measured by the expected long-term return. This reliance on data that it gathers by itself differentiates RL agents from those performing either supervised or unsupervised learning, and it is often a limiting factor in terms of the agent's ultimate success at mastering the environment. Specifically, if the agent only manages to visit a limited portion of the environment, its knowledge will be limited, leading to sub-optimal decision making (Wiering, 1999). However, if the agent focuses on acquiring information regarding parts of the environment that it has not seen sufficiently, it can lose the chance of gaining immediate reinforcement. This problem is referred to as *exploration-exploitation* trade-off, and is a crucial open problem in reinforcement learning (alongside generalization). Handling this trad-eoff is influenced by several factors, including the dynamics of the environment (*i.e.* transition probability and reward distribution), properties of the state/action spaces (*e.g.* discrete/continuous, number of states/actions, . . .), and the available number of interactions with the environment that the agent is allowed while it is training.

Our goal in this survey is to provide a broad high-level overview on the types of exploration methods employed by RL agents, by reviewing literature from the last three decades. Exploration studies have evolved during this time from simple ideas such as pure randomization, to increasingly effective methods which have interesting theoretical guarantees, or impressive empirical performance in large problems.

Exploration techniques have been categorized generally into *undirected* and *directed* methods (Thrun, 1992) based on the choice of information considered by the exploration algorithm. While in undirected exploration strategies, reinforcement learning agents select exploratory actions at random, without using *any exploration-specific knowledge*, directed exploration methods use the obtained information to pursue the exploration of less- visited state-action pairs, or of state-action pairs that are deemed to be more informative for the agent. However, this is not the only relevant differentiation between current exploration methods, as the field has expanded drastically and more nuances have developed.

In this survey, we present a more detailed categorization of exploration methods in reinforcement learning (see Figure 1). We attempt to group existing algorithmic approaches into these categories in order to provide a more detailed understanding of the current landscape of methods, in terms of both the goals and the information they employ. Of course, a few of the methods do not fall squarely into one category, and hence are included in multiple categories.

We note that some of the discussed techniques in this survey were originally proposed and designed for bandit settings, and only later applied in the reinforcement learning problems. However, this survey is not intended to cover exploration methods specifically designed for bandits, as many references exist in this area, including the excellent recent textbook Lattimore and Szepesvári (2020). We will focus only on sequential decision making, where exploration has an even larger impact, as it controls not only the immediate information received by the agent, but also the potential interesting information in its future data stream. In addition, the very large number of publications in this field, especially in recent years, requires making hard choices regarding which papers to discuss in this survey. In some cases, we have picked particular representative methods for certain approaches, rather than listing all instances of a particular approach. Finally, in order to provide a concise and easy-to-understand overview of the categories and methods, we do not focus on the mathematical

2

details and theoretical results in the field. Our main goal is to provide an entry point into the field, which is accessible to readers who may want to understand the types of algorithms and empirical evaluations that have been provided, and to practitioners who want to build successful applications, and hence have to tackle this really difficult problem.

The survey is organized as follows. In the following section, the notation is introduced and a brief RL background is provided. The exploration categories are subsequently presented in Section 3. Section 4 presents exploration methods that do not use reward information at all. Section 5 presents methods that rely mainly on randomizing action choices. Section 6 presents methods that are based on optimism in the face of uncertainty. Section 7 introduces methods which start with the formulation of the optimal exploration-exploitation trade-off, and then approximate its solution. Section 8 discusses probability matching methods, including posterior sampling. Finally, Section 9 provides some conclusions and perspectives.

2. Notation and Background

In reinforcement learning problems, at each discrete time step $t = 0, 1, 2, \ldots$, the system is at state s_t and the agent interacts with the environment by selecting action a_t. Consequently, the system transitions from the state s_t to s_{t+1}, determined by the transition model of the system $\mathcal{P}$ and returns a numerical reward r_t. Upon receiving the reward, the agent decides on the next action a_{t+1} and the process continues. Depending on the type of the problem, the decision making process either stops when it reaches a terminal state in an *episodic task* or continues in a *continuing task* with an infinite horizon. In the following paragraphs, we introduce the commonly used notation in this survey. Note that capital and calligraphic letters denote random variables and sets, respectively, unless stated otherwise. For a comprehensive introduction to reinforcement learning, refer to Sutton and Barto (1998a).

2.1 Markov Decision Processes

In this survey, the problems are modeled as Markov decision processes (MDPs) $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}, r \rangle$, where $\mathcal{S}$ and $\mathcal{A}$ denote sets of all possible states and actions in the system, respectively. MDPs assume that the environment is Markovian; *i.e.* the transition probability distribution $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \mathcal{D}(\mathcal{S})$ determines the next state from the probability distribution over the set of states $\mathcal{D}(\mathcal{S})$ as a function of the *current* state-action pair only. In other words, in MDPs we have,

$$\mathbb{P}\left(S_{t+1} = s' \mid s_t, a_t\right) = \mathbb{P}\left(S_{t+1} = s' \mid s_t, a_t, s_{t-1}, a_{t-1}, \ldots, s_0, a_0\right). \tag{1}$$

If the agent starts from state s and takes an action a, it transitions to the state s' with the probability,

$$P\left(s, a, s'\right) = \mathbb{P}\left(S_{t+1} = s' \mid S_t = s, A_t = a\right). \tag{2}$$

The expected rewards can be written as the reward function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, which maps the current state-action pair (s, a) to the immediate reward obtained from the set of real numbers $\mathbb{R}$ in the system,

$$r\left(s, a\right) = \mathbb{E}\left[R_{t+1} \mid S_t = s, A_t = a\right], \tag{3}$$

where the expectation is with respect to the randomness induced by the reward function r.

All the sequence of observations, actions and any kind of information the agent obtains during its lifetime is called history,

$$H_t = S_0, A_0, R_0, S_1, A_1, R_1, \ldots, S_t, A_t, R_t. \tag{4}$$

A trajectory τ is defined as the sequence of information extracted from the history H_T for the horizon T.

2.2 Reinforcement Learning Setting

In reinforcement learning setting, agent behaves according to a policy $\pi \in \Pi$, where Π represents the set of all possible policies. A deterministic policy $\pi : \mathcal{S} \to \mathcal{A}$ at time step t, also represented as $a_t = \pi(s_t)$, returns a particular action a_t, while a stochastic policy $\pi(a_t \mid s_t)$ gives a probability distribution over a set of actions ($\pi : \mathcal{S} \to \mathcal{D}(\mathcal{A})$), defined as

$$\pi(a_t \mid s_t) = \mathbb{P}(A_t = a_t \mid S_t = s_t). \tag{5}$$

The ultimate goal of reinforcement learning is to find an optimal policy $\pi^\star \in \Pi$, which maps states to actions that lead to the maximization of the expected (discounted) cumulative future reward J^π

$$J^\pi = \mathbb{E}_{\mathcal{P},\pi}[G], \tag{6}$$

where G denotes the *return*. For continuing tasks with infinite horizon, where the task never ends, G is defined as the discounted cumulative reward

$$G = \sum_{t=0}^{\infty} \gamma^t R_t, \tag{7}$$

where $\gamma \in [0, 1)$ is the discount factor, and is used to determine and control the importance of the future rewards. In the cases with finite horizon T (episodic tasks), the return G can be modified to the undiscounted version

$$G = \sum_{t=0}^{T} R_t. \tag{8}$$

The expected return for action selection policy π given the initial state $S_0 = s$, is called the state value function $V^\pi(s)$, written as

$$V^\pi(s) := \mathbb{E}_{P,\pi}[G \mid S_0 = s] = \mathbb{E}_{P,\pi}\left[\sum_{t=0}^{\infty} \gamma^t R_t \mid S_0 = s\right], \tag{9}$$

and is related to the expected return as $J^\pi = \mathbb{E}_P[V^\pi(S)]$.

An alternative to state value function is action value function $Q^\pi(s, a)$, which considers state-action pair (s, a) instead of state s only. In fact, it gives the value of taking action a in state s under a given policy π, and is defined as

$$Q^\pi(s, a) = \mathbb{E}_{P,\pi}[G \mid s, a] = \mathbb{E}_{P,\pi}\left[\sum_{t=0}^{\infty} \gamma^t R_t \mid s, a\right]. \tag{10}$$

Among all possible true value functions V^π for different policies $\pi \in \Pi$, there exists an optimal value function $V^\star$ corresponding to an optimal policy $\pi^\star$, which is at least as large as the others defined as,

$$V^\star(s) := \max_\pi V^\pi(s) \tag{11}$$

for all $s \in \mathcal{S}$. The optimal policy $\pi^\star(s)$ for all $s \in \mathcal{S}$ is thus a solution to $\max_\pi V^\pi(s)$.

Similarly, an optimal action value function $Q^\star$ can be defined for taking action a while being in state s. The optimal state value function $V^\star$ and optimal action value function $Q^\star$ are related as

$$V^\star(s) = \max_a Q^\star(s, a). \tag{12}$$

The optimal policy $\pi^\star(s)$ can thus be written as

$$\pi^\star(s) = \arg\max_a Q^\star(s, a). \tag{13}$$

Here, we introduced the general notation used throughout this survey. The notation specific to each section will be introduced in its respective category. In the next section, we propose a method of categorization for exploration techniques in reinforcement learning.

3. Categorization of Exploratory Techniques

Efficient exploration has been acknowledged as an important problem in adaptive control for quite a few decades, starting with the literature on bandit problems, eg. Thompson (1933). In this survey, however, we will not discuss the bandit literature, which is vast and has been the topic of the recent book Lattimore and Szepesvári (2020). Instead, we focus on sequential decision making. Some of the early studies that acknowledged the importance of efficient exploration in this context were delivered by Mozer and Bachrach (1990); Sutton (1990); Moore (1990); Schmidhuber (1990) and Barto et al. (1991). A study by Mozer and Bachrach (1990) showed that efficient learning of tasks modeled by a finite-state automaton is achievable using simple random exploration, provided that the number of states is small enough, while learning more complex tasks requires a more intelligent exploration technique that accelerates the coverage of the state space. In another study, Sutton (1990) demonstrated that the selection of sub-optimal actions (exploration) in non-stationary maze domains is essential for efficient learning, even though it may negatively affect the short-run acquisition of rewards.

Exploration techniques have been categorized in a few studies mainly based on the choice of inclusion of information in pursuing exploration. For instance, in a technical report, Moore (1990) emphasizes the interplay between the exploration-exploitation balance and efficient learning. He categorizes exploratory moves based on the action selection method into *entirely random*, *local random*, and *sceptical* categories. His proposed categorization states that while in the entirely-random method the agent chooses the exploratory actions totally at random, in the local-random experimentation it selects actions from the perturbed best known actions. As the very first action in the local-random method is chosen completely at random, the performance of this method is very sensitive to the quality of the first chosen action. Thus, a wrong decision at the initial step may lead to much larger learning times. The sceptical exploration method chooses actions depending on the prediction for the best

known action. If it is predicted to be unsuccessful, the agent explores other actions, and selects the best known action otherwise. Although this categorization of exploration methods can explain the similarities and differences between some of the early proposed approaches, it does not provide a general foundation for classifying exploration techniques.

One of the first *general* categorization of the exploration methods was introduced by Thrun (1992). He grouped exploration techniques into two general categories: *undirected* and *directed* methods. The undirected or *uninformed* exploration methods do not use any sort of exploration-specific knowledge in order to perform the exploration task. These methods generally rely *solely* on randomness in selecting actions. Random walk is the simplest method in this category, which is believed to be first utilized in action selection mechanisms by Anderson (1986), Munro (1987), Mozer and Bachrach (1990), Jordan (1989), Nguyen and Widrow (1990) and Thrun et al. (1991). Other examples of undirected methods consist of the exploration techniques related to Boltzmann distribution (based on utility and temperature parameter for controlling the exploration-exploitation trade-off) (Barto et al., 1991; Watkins, 1989; Lin, 1992; Singh, 1992; Sutton, 1990; Lin, 1990) and random action selection with a certain probability (Whitehead and Ballard, 1991; Mahadevan and Connell, 1992, 1991). On the contrary, directed or *informed* exploration methods utilize exploration-specific knowledge of the learning process to direct the agent towards exploring the environment. These exploration techniques are more efficient and beneficial compared with the undirected exploration methods in terms of complexity and cost (Thrun, 1992; Whitehead, 1991). In particular, random exploration methods may lead to an increase in the learning time as well as safety issues due to the random selection of unsafe actions repeatedly (especially in real cases, such as in robotics). Consequently, in the following years, exploration-related studies focused increasingly on "directed" exploration methods. The consequent diversity in these approaches necessitates the provision of a new basis for a refined categorization of these methods.

To address the issues corresponding to the existing categorization methods, we propose a new classification approach based on the type of information the agent uses to explore the world (Figure 1). In particular, we categorize the RL exploration methods into the two general classes, namely "Reward-Free Exploration", where the included exploration techniques do not use the extrinsic rewards in their action selection process, and "Reward-Based Exploration", in which extrinsic rewards affect the choice of exploratory actions. These two classes are further divided into two groups "Memory-Free" and "Memory-Based" techniques, depending on the reliance of the exploratory movements on the agent's memory of the observed space. The categories in each class of exploration techniques are described below and detailed in the following sections.

- **Reward-Free Exploration** - The general property of the exploration methods included in this category is that rewards (or value functions) do not affect the choice of actions in their action-selection criteria. In other words, actions are selected without regard to the obtained rewards or the value functions. The methods belonging in this section can either act completely blindly, which we call *blind exploration*, or utilize some sort of information (other than extrinsic rewards) in the form of intrinsic rewards in order to encourage exploration. This type of exploration methods are referred to
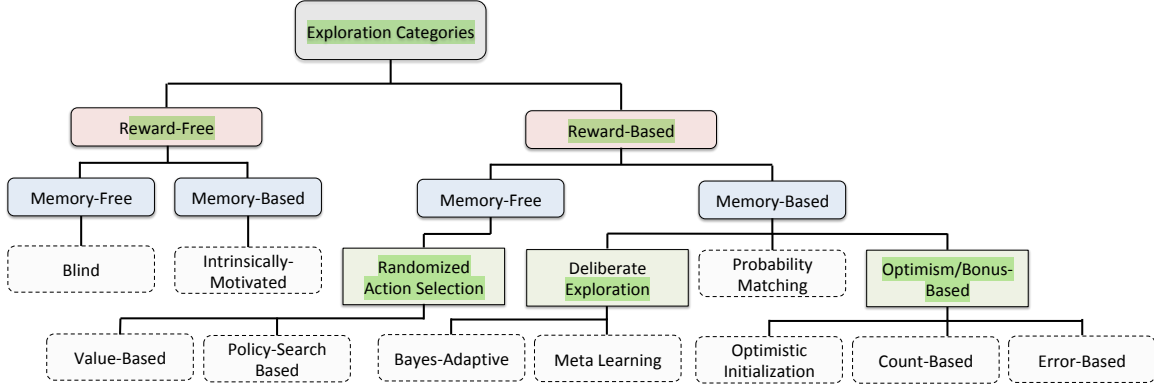
Figure 1: Exploration Categories- The exploration methods are categorized into two main groups *reward-free* and *reward-based* exploration techniques, depending on their utilization of extrinsic rewards. Each group is further divided to *memory-based* and *memory-free* categories based on the reliance of the exploratory decisions on the agent's memory of the observed space.

as *Intrinsically-Motivated Exploration* techniques. The details regarding this category and its subcategories are provided in section 4.

- **Randomized Action Selection -** The exploration methods in this category induce exploratory behaviour via assigning action selection probabilities to the admissible actions based on the estimated value functions or rewards (*Value-Based Exploration*), or the learned policies (*Policy-Search Based Exploration*). The exploration methods included in the former group use the reward-based feedback in order to handle the exploration-exploitation trade-off. The list of the exploration techniques in this category as well as the detailed explanation of each method are provided in section 5.1. In the latter group, exploration methods explore the environment via performing search in the space of policies. They learn a stochastic policy, whose stochasticity helps the agent balance the trade-off between exploration and exploitation in the system. These methods explicitly represent policies, and aim to update them to maximize the expected extrinsic rewards (section 5.2). Note that the policy-search based methods that *do not* utilize extrinsic rewards in their exploratory decision making are listed and discussed in section 4.

- **Optimism/Bonus-Based Exploration -** The exploration techniques in this category function based on the principle of *optimism in the face of uncertainty*, where actions with uncertain values are preferred over the rest of the possible actions. In this category of exploration methods, the methods usually involve a form of *bonus*, which is added to the extrinsic reward, leading to a directed search in the spaces of state-action. The methods included in this category and the details are provided in section 6. The main difference between bonus-based techniques and the intrinsically-motivated exploration methods, discussed in section 4, is that the latter does *not* utilize extrinsic reward for motivating the exploration of the environment. Different forms

of optimism/bonus-based exploration approaches are discussed in section 6, including *count-based* exploration methods and *prediction-error based* approaches. Finally, the clear distinction between the methods in this category and those in the *Stochastic Action Selection* methods (5) is that the techniques considered as *Optimism/Bonus-Based* direct the agent's moves with the use of *bonuses*, while the methods in the other group rely solely on the extrinsic rewards.

- **Deliberate Exploration -** This category includes exploration methods that operate based on solving the exploration-exploitation tradeoff optimally and is discussed in section 7. This category consists of *Bayes-Adaptive* exploration methods that are realized with a Bayesian model-based set-up, where the *posterior* distribution over models is computed and updated assuming a prior over the transition dynamics. This group also consists of *Meta-Learning Based Exploration* techniques, via which the agent learns to adapt quickly using the prior given tasks.

- **Probability Matching -** This category of exploration techniques uses a heuristic to decide the next action based on sampling a single instance from the posterior belief over environments or value functions, and solving for that sampled environment exactly. The agent can then act according to that solution, e.g. for the duration of one episode. Each action is thus taken with the according to the probability the agent considers it to be the optimal action. This heuristic effectively directs exploration effort to promising actions.

In terms of the categorization proposed by Thrun (1992), we can classify our proposed categories into the two general groups of *directed* and *undirected* exploration techniques. In this regard, in the reward-free exploration category, *blind* exploration methods (section 4.1) are undirected, while the *intrinsically-motivated* exploration techniques (section 4.2) are considered directed exploration approaches. The *stochastic action selection* exploration category (section 5.1) consists of undirected techniques. The rest of the categories fall under the directed exploration category.

In the following sections, the above mentioned groups of exploration techniques are discussed in more detail and the methods under each group are explained. In a few cases among the exploration techniques, there exist some approaches that belong to two of the proposed categories, leading to a small overlap between the groups. Whenever such situation is encountered, the respective exploration method is noted as shared between the corresponding categories.

4. Reward-Free Exploration

We use the notion of *reward-free* exploration in reinforcement learning to describe any method of exploration that does not incorporate extrinsic reward in their exploratory action selection criteria. This type of exploration methods was first introduced and utilized with the name *pure exploration* in multi-armed bandits, a set of sequential decision-making tasks where at each time step, an agent pulls an arm and receives a random reward drawn from the reward distribution of that specific arm (Bubeck et al., 2009). In particular, these exploration techniques do not incorporate the rewards obtained from the environment

(*i.e.* extrinsic rewards) in measuring the cost of picking bandit arms. Instead, they utilize other available resources in a limited budget, such as CPU time or cost, in order to acquire knowledge. Similarly, there have been reward-free exploration methods proposed for the reinforcement learning framework, which do not rely on the extrinsic rewards the agent receives from the environment. These exploration techniques can be: 1) completely blind, where the exploratory agent selects actions in the absence of any sort of information obtained as the result of its interaction with the environment; or 2) driven by some form of intrinsic motivation and curiosity. These two forms of reward-free exploration methods are further explained, and the corresponding proposed methods are discussed in the following paragraphs and listed in Table 1. Note that in neither of the aforementioned cases, the agent uses extrinsic reward as a source of knowledge. Thus, bonus-based methods (*i.e.* exploration methods that rely on a sort of bonus reward in *addition to* extrinsic reward) do not belong in this category and are discussed in *Optimism/Bonus-Based Exploration* section (section 6).

4.1 Blind Exploration

Exploration techniques in the *blind exploration* category explore environments solely on the basis of random action selection. In other words, these agents are not guided through their exploratory path by any form of information, thus are uninformed or *blind*. This category of exploration techniques is indeed the most basic type of reward-free exploration, and includes random-walk as the simplest exploration method (Thrun, 1992). Some examples of the early uses of random walk in exploring the effect of various actions on different states were the studies performed by Anderson (1986); Mozer and Bachrach (1990) and Jordan (1989). In the random walk method, the agent chooses actions randomly regardless of the information it has obtained so far and thus, due to the uniformly random probability of selecting actions, there is a chance that the picked action takes the agent away from the goal rather than taking it closer (*i.e.* exploration). On the other hand, it leads to large complexity that grows exponentially with the size of the environment (Whitehead, 1991), which makes random-walk an inefficient exploration technique.

Another simple yet effective exploration method in this category is known as the method of ϵ-greedy (Sutton, 1996), also known as *max-random* or *pseudo-stochastic* (Caironi and Dorigo, 1994; Watkins, 1989). In the ϵ-greedy approach, the parameter $\epsilon \in [0, 1]$ controls the balance between exploration and exploitation. The action a_t at every time step t is chosen such that,

$$a_t = \begin{cases} a_t^\star, & \text{with probability 1-}\epsilon \\ random\ action & \text{with probability } \epsilon, \end{cases} \tag{14}$$

where $a_t^\star$ is the greedy action taken at time t with respect to the greedy policy (exploitation).

The ϵ-greedy method has been found to be quite effective in different RL settings (Sutton and Barto, 1998b). In particular, it is efficient in the sense that it does not need to cache any data to perform exploration, and the only hyperparameter to adjust is ϵ. However, despite the fact that the ϵ-greedy method guarantees that at infinite time horizon every state-action pair is visited infinitely often, it stays sub-optimal in the sense that it asymptotically prevents the agent from selecting the best action (Vermorel and Mohri, 2005). Another problem an

9

agent may encounter while using ϵ-greedy is the lack of decisiveness in the exploration phase, which might in turn lead to getting stuck in local optima. To address this issue, a temporally extended form of ϵ-greedy, called ϵz-greedy (Dabney et al., 2020), has been proposed, where random exploratory actions are replaced by temporally-extended sequence of actions. In particular, the ϵz-greedy agent exploits with probability $1 - \epsilon$ and explores via repeating the same action for a certain number of steps $n \sim z$, where $z(n)$ is a distribution over the action-repeat duration n. Dabney et al. (2020) perform experiments in tabular as well as deep RL frameworks in tasks with discrete-action spaces, and tabular or discretized continuous state spaces.

Although ϵ is usually hand-tuned depending on the type of the problem, there are other proposed extensions of the ϵ-greedy method, such as ϵ-first (Tran-Thanh et al., 2010) (where exploration is done during the first ϵT time steps- T is the total number of steps) and decreasing-ϵ in bandits (Caelen and Bontempi, 2007), where ϵ is a decreasing function of *time*, as well as the derandomization of ϵ-greedy in RL tasks (Even-Dar and Mansour, 2002). Another extension of the ϵ-greedy approach is the *Value-Difference Based Exploration* (VDBE) method (Tokic (2010) in Bandits and Tokic and Palm (2011) in reinforcement learning), which adjusts the exploration rate ϵ based on the changes in the state-action value functions. The methods Even-Dar and Mansour (2002); Tokic (2010); Tokic and Palm (2011), which incorporate extrinsic rewards in their exploratory decision making, are discussed in detail in section 5.1. There are other blind exploration methods mainly proposed in the field of Robotics, for instance the spiral search technique (Burlington and Dudek, 1999), which ensures visiting new locations in planar environments via expanding the search in the space with the use of logarithmic spirals. However, due to the study limit of these methods to Robotics and planar environments, we are not going to cover them here.

4.2 Intrinsically-Motivated Exploration

The second exploration type in the *reward-free exploration* category is the *intrinsically motivated exploration,* which is composed of the methods that utilize a form of intrinsic motivation in the absence of external rewards to promote exploring the unexplored parts of the environment. In contrast to blind exploration, intrinsically-motivated exploration techniques utilize some form of intrinsic information to encourage exploring the state-action spaces. The idea of employing internal incentives in exploratory tasks is borrowed from *intrinsically-motivated behaviour* in humans, which has been studied and discussed extensively in education and psychology literature (Deci, 1971, 1975; Amabile et al., 1976; Benware and Deci, 1984; Deci and Ryan, 1985; Grolnick and Ryan, 1987). In psychology, the distinction between an extrinsically and an intrinsically motivated behaviour is made based on the types of the stimuli that "move" the person to perform a task (Ryan and Deci, 2000). While intrinsic motivation leads to an inherent satisfaction of performing a job, extrinsic motivation sets an external regulation, which encourages a person to do a task in order to obtain some separable outcome (*e.g.* reward or reinforcement). Studies show that internalization of the external regulations, also known as self-regularization or self-determination (Deci and Ryan, 1985), helps children attain higher achievements (Benware and Deci, 1984; Grolnick and Ryan, 1987) in terms of learning or completing complex tasks, in contrast to using extrinsic motivations in the form of rewards or reinforcement. Analogs of intrinsic motivation

in human can be employed in order to promote exploration in the RL framework. Here, we review some of these studies and discuss different forms of intrinsic motivation that have been used in exploration techniques in the reinforcement learning tasks.

In the context of reinforcement learning, curiosity takes various interpretations and forms depending on the types of problems and the defined goals as well as the approaches taken toward understanding and solving the problems. In general, we can define curiosity as a way or desire to explore new situations that may help the agent with pursuing goals in the future. As agents are encountered with deceptive and/or sparse rewards in many RL set-ups, exploratory agents that rely on extrinsic rewards might end up in *local optima* because of deceptive rewards or get stuck due to zero gradient in the received rewards. Thus, the techniques that intrinsically motivate the agent to explore the environment and do not rely on the extrinsic reinforcement are effective in learning of such tasks.

Many of the reward-free intrinsically-motivated exploration strategies aim at minimizing the agent's *uncertainty* or *error* in its predictions (Schmidhuber, 1991a,b; Pathak et al., 2017). In order to evaluate the precision of the agent's predictions of the environment behavior, a model of the environment dynamics is required, such that given the current state and the chosen action, the model predicts the next state. Minimization of the resulting error in the model prediction encourages the exploration of the underlying space. There are other reward-free techniques that pursue the maximization of *space coverage* (new states or state-action pairs visitation), which utilize a form of intrinsic motivation to govern the agent's exploratory behaviour (Hazan et al., 2019; Amin et al., 2020). More *space coverage* essentially means visiting more unexplored states in a shorter amount of time and in turn, learning more about the environment. In this section, we survey and discuss some of the reward-free intrinsically-motivated exploration approaches that seek either of the above-mentioned goals. Note that the notion of *intrinsic motivation* in RL tasks has been also used in combination with external rewards, which is not the subject of our discussion in this section and will be elaborated in the "Bonus/Optimism-Based Exploration" category (section 6).

The early use of intrinsic motivation in computational framework dates back to 1976, when Lenat (1976) used the notion of "interestingness" in mathematics to encourage new concepts and hypotheses. Scott and Markovitch (1989) introduced DIDO, a curiosity-driven learning strategy that can help the agent explore initially unknown domains in an unsupervised set-up using the notion of Shannon's uncertainty function defined as

$$sh = -\sum_{i=1}^{n} \left(p_i \times \log_2(p_i) \right), \tag{15}$$

where p_i is an estimate of the probability of the outcome O_i, and the summation is taken over all outcomes. Minimization of *uncertainty* in their proposed formalism leads to a broader searching span, as well as more effective learning of the search space. DIDO, in fact, promotes the idea of using an experience generator that provides experiences, which are novel compared to the previous ones and are related to them at the same time. The obtained experiences help the agent search for a better representation in the space of possible representations while DIDO's representation generator is employed in finding more informative experiences. Scott and Markovitch (1989) performed DIDO in several discrete

domains, which showed that their exploratory algorithm enables the agent to select sensible experiences and eventually leads to a good representation of the domain.

In the following years, another form of curiosity-driven exploration was introduced (Schmidhuber, 1991a,b) based on the improvement in the reliability of the RL agent's predictions of the world model. In particular, Schmidhuber (1991b) proposed a model-building control system that could provide an adaptive model of the environmental dynamics. He further proposed the notion of dynamic *curiosity* and *boredom*, described as "the explicit desire to improve the world model", as a potential means of increasing the knowledge of the animat about the world in the exploration phase. In his work, *curiosity* aims at minimization of the agent's ignorance and is triggered when the agent comes to the realization that it does not have enough knowledge of something. It provides a source of reinforcement for the agent and is defined as the Euclidean distance between the real and the predicted model network. A failure in the correct prediction of the environment leads to a positive reinforcement that encourages the agent to further explore the corresponding actions. Improvement in the world model predictions with time leads to less reinforcement and thus, discouragement of exploring the corresponding actions, also referred to as *boredom*.

While the idea of using curiosity was not implemented in Schmidhuber (1991b), Schmidhuber later utilized the notion of *adaptive curiosity* (Schmidhuber, 1991a) to encourage exploration of the unpredictable parts of the environment. In particular, he proposed a curious model-building control system, where the notion of *adaptive confidence* was used for modeling the reliability of a predictor's predictions, and *adaptive curiosity* was utilized to reward the agent for encountering hard but learnable states and thus improve the *exploration* phase by reducing the extra time spent on the non-useful or well-modelled parts of the environment. The strength of his proposed approach in comparison to its predecessors including his previous study (Schmidhuber, 1991b), lies in its ability to work in *uncertain non-deterministic* environments by adaptively modeling the reliability of a predictor's predictions and learning to predict *cumulative* error changes in the model network. This goal can be achieved via maximization of the expectation of cumulative changes in prediction reliability. Schmidhuber (1991a) tested his proposed curiosity-driven algorithm based on Watkin's Q-learning in two-dimensional *discrete-state* toy environments with over 100 states and compared the results to the ones obtained using random search as the exploratory approach. Utilizing an adaptive curious agent led to a decrease of an order of magnitude in the learning time.

Another similar yet different exploration approach was proposed by Thrun and Möller (1992) around the same time, which suggested using the notion of *competence map* for guiding exploration via estimating the controller's *accuracy*. In particular, Thrun and Möller (1992) introduced a notion of energy

$$E = (1 - \Gamma)E_{\text{explore}} + \Gamma E_{\text{exploit}}, \tag{16}$$

where gain parameter $0 < \Gamma < 1$ controls the exploration-exploitation trade-off and is a function of the change in the exploration energy E_{explore} and the exploitation energy E_{exploit}. A competence network system is trained to estimate the upper bound of the "model network error"; minimization of the "expected competence" leads to the exploration of the world. Thrun and Möller (1992) employed "competence map" in a continuous two-dimensional

robot navigation task, which revealed that their suggested exploration method can perform better compared with "random walk" and "purely greedy" approaches.

An exploration approach was later proposed by Storck et al. (1995) as an extension of previous similar studies, such as Schmidhuber (1991a,b); Thrun and Möller (1992). Their proposed exploration method, called *Reinforcement Driven Information Acquisition* (RDIA), is devised for non-deterministic environments and utilizes the notion of *information gain*, which is used as an intrinsic motivation to govern the agent's exploratory movements. In particular, the RDIA agent models the environment via estimating the transition probability $p_{ijk}^\star(t)$ at each time step t as the ratio of the number of times so far that the pair (s_i, a_j) has led to the state s_k over the number of times the agent has experienced (s_i, a_j). The information gain is then defined as the difference between the agent's current estimation of the transition probability $p_{ijk}^\star(t)$ and $p_{ijk}^\star(t+1)$ at time $t+1$. Information gain represents the information that the agent has acquired upon performing the respective action, which consequently leads to an increase in the estimator's accuracy. Storck et al. (1995) assess their proposed method in simple discrete environments with certain numbers of states and actions using two different information gain measures, namely the entropy difference and the Kullback-Leibler (KL) distance, between the probability distributions, and show that the results obtained by RDIA surpass those of simple random search.

Information gain as intrinsic motivation has been used in other exploration strategies such as the studies performed by Little and Sommer (2013) and Mobin et al. (2014). In particular, the exploration method proposed by Little and Sommer (2013) takes a Bayesian approach, where the agent builds an internal model of the environment, and upon taking an action and observing the next state, calculates the KL-divergence of its current internal model from the one it had predicted prior to taking the action. The resulting unweighted sum of the KL-divergences yields the *missing information* I_M, which is utilized as a measure of inaccuracy in the agent's internal model. The agent subsequently takes actions that maximize the expected (predicted) information gain (PIG), defined as the expected decrease in the missing information I_M between the internal models. Another similar exploration method (Mobin et al., 2014) extends the application of PIG (Little and Sommer, 2013) to perform in environments with *unbounded* discrete state spaces. In particular, Mobin et al. (2014) utilize the Chinese Restaurant Process (CRP)(Aldous, 1985) to find the probability of revisiting a state or discovering a new one, and use the obtained results to calculate the agent's internal model. The subsequent steps are similar to those presented and discussed by Little and Sommer (2013). Another study by Shyam et al. (2019) introduces the Bayesian Model-based Active eXploration (MAX) method, which utilizes the *novelty of transitions* as a learning signal, and is applicable in discrete and continuous environments. In particular, MAX agent calculates the Jenson-Shannon divergence and the Jensen-Rényi divergence (Rényi et al., 1961) of the predicted space of distributions from the resulting one in discrete and continuous environments, respectively. Maximization of the resulting novelty measure governs the agent's exploratory behaviour. The evaluation of MAX performance in several discrete and continuous tasks presents promising results compared with those obtained from MAX counterparts and other baselines.

Another curiosity-driven approach is "Intrinsic Curiosity Module" (ICM) (Pathak et al., 2017), where curiosity is defined as "the error in an agent's ability to predict the consequence of its own actions". In their set-up, the agent interacts with high-dimensional continuous

Approach	Intrinsic Motivation	Remarks
Anderson (1986)	None (Blind)	Early use of *random walk*
Mozer and Bachrach (1990)	None (Blind)	Early use of *random walk*
Jordan (1989)	None (Blind)	Early use of *random walk*
Sutton (1996)	None (Blind)	ϵ-greedy
Caironi and Dorigo (1994)	None (Blind)	max-random (ϵ-greedy)
Dabney et al. (2020)	None (Blind)	ϵz-greedy (Temporally-extended actions)
Burlington and Dudek (1999)	None (Blind)	Spiral search (For planar environments only)
Schmidhuber (1991b)	Uncertainty	Adaptive model of the environment dynamics
Scott and Markovitch (1989)	Uncertainty	Minimization of Shannon's uncertainty function
Schmidhuber (1991a)	Uncertainty	Prediction of cumulative error changes
Thrun and Möller (1992)	Uncertainty	Competence map
Storck et al. (1995)	Uncertainty	Maximization of information gain
Little and Sommer (2013)	Uncertainty	Maximization of expected information gain
Mobin et al. (2014)	Uncertainty	Maximization of expected information gain
Shyam et al. (2019)	Uncertainty	Uses an ensemble of forward dynamics models
Pathak et al. (2017)	Uncertainty	Minimization of predicted error in feature representation
Hazan et al. (2019)	Space coverage	Maximization of entropy of the distribution over the visited states
Amin et al. (2020)	Space coverage	Generation of correlated trajectories in the state and action spaces
Forestier et al. (2017)	Self-generated goals	Coverage maximization of the space of goals
Colas et al. (2018)	Self-generated goals	Combines Forestier et al. (2017) with DDPG
Machado et al. (2017)	Space coverage	Maximization of eigenpurposes
Machado et al. (2018b)	Space coverage	Extension of Machado et al. (2017) to stochastic environments
Jinnai et al. (2019)	Space coverage	Minimization of cover time
Jinnai et al. (2020)	Space coverage	Extension of Jinnai et al. (2019) to large or continuous state spaces
Hong et al. (2018)	Space coverage	Encouraging new policies using a distance measure between the policies

Table 1: Examples of some reward-free exploration approaches.

state spaces (images in this case). The authors show that ICM helps the agent to learn and improve its exploration policy in the presence of sparse extrinsic rewards as well as in the absence of any sort of environmental rewards. Moreover, they show that the curious agent can apply its gained knowledge and skills in new scenarios and still achieve improved results. The main idea behind ICM is that instead of targeting *learnable* states and rewarding the agent for detecting them (Schmidhuber, 1991b,a), ICM (Pathak et al., 2017) focuses only on a *feature representation* that reflects the parts of the environment that either affect the agent or get affected by the agent's choice of actions. Intuitively, by focusing on the influential feature space instead of the state space, ICM is able to avoid the unpredictable or unlearnable parts of the environment.

Note that in some of the fore-mentioned proposed algorithms (Schmidhuber, 1991b,a; Pathak et al., 2017), while the external reinforcement is not a necessary component, the external reward, if exists, can be added to the curious reinforcement. Thus, these studies are included in the list of "Bonus/Optimism-Based Exploration" category as well (section 6).

Some of the intrinsically-motivated exploration techniques utilize the analogy between the dynamical and physical systems, and thus propose the notion of *entropy maximization* to encourage exploration of the search space. In this regard, an early utilization of entropy in intelligent adaptation of search control was in the context of *search effort allocation* problem in genetic search procedures (Rosca, 1995), where entropy was used as a measure of diversity. Around the same time, Wyatt (1998) defined a notion of entropy for the case of bandits as a measure of the uncertainty regarding the identity of the optimal action. In his proposed exploration algorithm, the agent selects the more informative action, which is the one that on expectation will lead to a larger entropy reduction. In the field of reinforcement learning (which is the main focus of the current survey), there are several studies that utilize notion of entropy in their proposed exploration techniques (Achbany et al., 2006; Lee et al., 2018; Yin, 2002; Hazan et al., 2019). In this section, however, we discuss the method proposed by Hazan et al. (2019) as it is the only *reward-free* technique among the studies that utilize the notion of entropy in guiding exploration. Hazan et al. (2019) introduce an exploration approach, which targets environments that do not provide the agent with extrinsic reward. In their proposed method, the intrinsic objective is to maximize the entropy of the distribution over the visited states. They introduce an algorithm, which optimizes objectives that are only functions of the state-visitation frequencies. In particular, it generates and optimizes a sequence of intrinsic reward signals, which consequently leads to the entropy maximization of the distribution that the policy induces over the visited states. The reward signals form a concave reward functional $R(d_\pi)$, which is a function of the entropy of the induced state distribution d_π given policy π. The obtained optimal policy is referred to as *maximum-entropy* (MaxEnt) exploration policy, which is defined as $\pi^\star \in \arg\max_\pi R(d_\pi)$.

Another intrinsically-motivated exploration approach that encourages space coverage is the method of PolyRL (Amin et al., 2020), which is designed for tasks with continuous state and action spaces and sparse reward structures. PolyRL is inspired by the statistical models used in the field of polymer physics to explain the behaviour of simplified polymer models. In particular, PolyRL exploration policy selects orientationally correlated actions in the action space and induces persistent trajectories of visited states (*locally self-avoiding*

walks) in the state space using a measure of spread known as the *radius of gyration squared*,

$$U_g^2(\tau_\mathcal{S}) := \frac{1}{T_e - 1} \sum_{s \in \tau_\mathcal{S}} d^2(s, \bar{\tau}_\mathcal{S}). \tag{17}$$

In equation 17, T_e denotes the number of exploratory steps taken so far in the current exploratory trajectory, $\tau_\mathcal{S}$ is the trajectory of the visited states, and $d(s, \bar{\tau}_\mathcal{S})$ is a measure of distance between a visited state s and the empirical mean of all visited states $\bar{\tau}_\mathcal{S}$. At each time step, the exploratory agent computes $U_g^2(\tau_\mathcal{S})$, and subsequently compares it with the value obtained from the previous step. In addition, it calculates the high-probability confidence bounds on the radius of gyration squared, within which the stiffness of the trajectory is maintained. If the change in $U_g^2(\tau_\mathcal{S})$ is within the confidence interval, the agent continues to explore, otherwise it selects the subsequent action using the target policy. Amin et al. (2020) assess the performance of PolyRL in 2D continuous navigation tasks as well as several high-dimensional sparse MuJoCo tasks and show improved results compared with those obtained from several other exploration techniques.

Policy-Search Based Exploration without Extrinsic Reward - *Policy-search* methods search in the parameter space θ for the appropriate parameterized policy π_θ. As policy-search methods typically do not learn value functions, the choice of a proper parameter θ is essential for ensuring an efficient, stable and robust learning. This calls for an efficient exploration strategy in order to provide the policy evaluation step in the policy search methods with new trajectories and thus new information, which is subsequently used for policy update (Deisenroth et al., 2013). The exploration approaches performed in policy search methods use stochastic policies, and they can either employ rewards obtained from the environment to guide the exploratory trajectories or function in a completely reward-free manner. In the current section, we review the policy-search methods that do *not* incorporate extrinsic rewards in their exploratory decision making. We provide a more thorough introduction to policy-search methods in section 5.2, where we discuss the policy-search approaches that utilize extrinsic rewards.

One of the approaches to solving problems autonomously is breaking the problem/goal into smaller sub-problems/sub-goals (Forestier et al., 2017). This idea is inspired from the way children tend to select their objectives such that they are not too easy or too hard for them to handle. These intermediate learned goals facilitate learning more complex goals, which ultimately lead to building up more skills required to achieve bigger goals. Based on this intuition, Forestier et al. (2017) propose a curiosity-driven exploration algorithm called "Intrinsically Motivated Goal Exploration Process" (IMGEP). The IMGEP approach structure relies on assuming that the agent is capable of choosing goal p from the space of RL problem and is able to calculate the corresponding reward r using the reward function $R(p, c, \theta, o_\tau)$ given the parameterized policy π_θ, context c (which gives the current state of the environment) and the observed outcome o_τ in the trajectory $\tau = \{s_{t_0}, a_{t_0}, s_{t_1}, a_{t_1}, \ldots, s_{t_{\text{end}}}, a_{t_{\text{end}}}\}$. The reward function $R(p, c, \theta, o_\tau)$ is thus non-Markovian and can be calculated at any time during or after performing the tasks. Using the computed rewards, the agent samples the *interesting* goal p, which is a self-generated goal that leads to faster learning progress. In the exploration phase, the agent uses the meta policy $\Pi_\epsilon(\theta|p, c)$ to find the parameter θ for goal p, which is subsequently utilized in a goal-

parameterized policy search process. The obtained outcome is then used in computing the intrinsic reward r, which in turn provides useful information regarding the interestingness of the samples goal p. The goal sampling strategy and the meta-policy are subsequently updated. The performance of IMGEP in the case of a real humanoid robot shows that the IMGEP robot can effectively explore high-dimensional spaces through discovering skills with increasing complexity.

One of the exploration methods proposed based on the "Goal Exploration Processes" (GEPs) (Forestier et al., 2017) is the "Goal Exploration Process- Policy Gradient" (GEP-PG) (Colas et al., 2018), which combines the intrinsically-motivated exploration processes GEPs with the deep reinforcement learning method DDPG in order to improve exploration in continuous state-action spaces and learn the tasks. Colas et al. (2018) perform GEP-PG in the low-dimensional "Continuous Mountain Car" and the higher-dimensional "Half-Cheetah" tasks. GEP-PG is tested in the fore-mentioned problems with different variants of DDPG. The authors show that specifically in the Half-Cheetah task, the performance, variability and sample efficiency of their proposed method surpasses those of DDPG.

Another policy-search based exploration strategy proposed by Machado et al. (2017) utilizes the notion of *proto-value functions* (PVFs) to discover *options* that lead the agent towards efficient exploration of the state space. PVFs, first introduced by Mahadevan (2005), are the basis functions used for approximating value functions through incorporating topological properties of the state space. In particular, using the MDP's transition matrix, a diffusion model is generated, whose diagonalized form subsequently gives rise to PVFs. The diffusion model provides the diffusion information flow in the environment. This feature allows the PVFs to provide useful information regarding the geometry of the environment, including the *bottlenecks*. Machado et al. (2017) define an intrinsic reward function (a.k.a. *eigenpurpose*) as,

$$r_{\text{in}}^{\mathbf{e}}\left(s, s'\right) = \mathbf{e}^{\mathsf{T}}\left(\phi\left(s'\right) - \phi\left(s\right)\right),\tag{18}$$

where $\mathbf{e} \in \mathbb{R}^{|\mathcal{S}|}$ is the proto-value function and $\phi(s)$ is the feature representation of state s, which can be replaced by the state s itself in tabular cases. Machado *et al.* utilize eigenpurpose $r_i^{\mathbf{e}}$ to discover options (*a.k.a.* eigenoptions) and their corresponding eigenbehaviors. They subsequently use policy iteration to solve the problem for an optimal policy.

Although the exploration technique introduced by Machado et al. (2017) is applicable to discrete domains only, one of its major advantages is that it provides a dense intrinsic reward function, which facilitates exploration tasks with sparse-extrinsic-reward structures. Moreover, since it is equipped with options, their proposed method can cover a relatively larger span in the state space compared with that of a simple random walk. Finally, the authors show that their method with options is effective for exploration tasks with the goal of maximizing the cumulative rewards. Later work by Machado et al. (2018b) proposed an improved version of *eigenoption discovery*, extending it to stochastic environments with non-tabular states. In particular, using the notion of successor representation Dayan (1993), in their proposed method the agent learns the non-linear representation of the states which in turn gives the diffusive information flow (DIF) model, and subsequently, the eigenpurposes and eigenoptions are obtained.

In the recent years, Jinnai et al. (2019) introduced the method of *covering options*, where options are generated with the goal of minimizing cover time. Their proposed method

encourages the agent to visit less-explored regions of the state space by generating options for those parts, without using the information obtained from extrinsic rewards. Their empirical evaluation in discrete sparse-reward domains present reduced learning time in comparison with that of some of their predecessors. The method of *deep covering options* (Jinnai et al., 2020) extends covering options to large or continuous state spaces while minimizing the agent's expected cover time in the state space. The authors have successfully shown the behaviour of deep covering options in challenging sparse-reward tasks, including Pinball, as well as some MuJoCo and Atari domains.

In a study by Hong et al. (2018), the authors apply a diversity-driven method to off- and on-policy DRL algorithms and improve their performances in large state spaces with sparse or deceptive rewards via encouraging the agent to try new policies. In particular, the agent uses a distance measure to evaluate the novelty of π in comparison to the prior ones and subsequently modifies the loss function,

$$L_D = L - \mathbb{E}_{\pi' \in \Pi'} \left[\alpha D \left(\pi, \ \pi' \right) \right], \tag{19}$$

where L denotes the loss function of the deep RL algorithm, α is a scaling factor, and D is a distance measure between the current policy π and the policy π' sampled from a set of most recent policies Π'. Equation 19, encourages the agent to try new policies and explore unvisited states without relying on the extrinsic rewards received from the environment. The agent will consequently overcome the problem of getting stuck in local optima due to the presence of deceptive rewards or failing to learn tasks with sparse rewards or large state spaces. The authors apply their proposed exploration method in 2D grid worlds with deceptive or sparse rewards, Atari 2600 as well as MuJoCo, and show that it enhances the performance of DRL algorithms through a more effective exploration strategy.

5. Randomized Action Selection

So far, we have introduced and discussed exploration methods that do not acquire information in the form of extrinsic rewards in the process of exploratory action selection. In this section and the rest of the survey, we focus on the exploration techniques that make decisions using extrinsic rewards with or without other forms of information obtained from the learned process. In this section, we specifically target the exploration methods that assign action selection probabilities to the admissible actions based on value functions/rewards or the learned policies. The two groups of exploration methods are introduced in sections 5.1 and 5.2, respectively.

5.1 Value-Based Methods

A simple way to deal with the exploration-exploitation trade-off is to induce exploratory behaviour via assigning action selection probabilities to the admissible actions based on the estimated value functions. In the early phase of learning, the agent should be able to try different actions in each state. Later in the intermediate learning phase, if the agent's target policy takes the control of the action selection process, it may lead to a partial visitation of the state space, and thus a sub-optimal policy and value function. To tackle this issue, there are exploration approaches that select the stochastic actions based on the feedback

they receive, in the form of value function or rewards, from the environment. Using these feedback, they balance exploration and exploitation via deciding between acquiring more knowledge from the environment and maximizing the obtained rewards. In this section, some of the exploration methods that perform action selection according to the above-mentioned criteria are discussed below. Examples of some value-based randomized action selection exploration methods are provided in Table 2.

One of the most important exploration methods in this category is the *Softmax* action selection method (Bridle, 1990). In this method, the greedy action at the current state is selected with the highest probability, while the other actions are given the probability of being selected according to their estimated values. The most commonly used formalism for performing Softmax action selection is the *Boltzmann* distribution. The early use of Boltzmann distribution in exploration was by Watkins (1989); Lin (1992) and Barto et al. (1991). In the Boltzmann exploration approach, the value function $Q_t(s,a)$ for the current state $S_t = s$ and action $a_t = a$ assigns the probability of selecting action a as,

$$\pi_t\left(a|s\right) = \frac{e^{Q_t(s,a)/T_m}}{\sum_{i=1}^{N} e^{Q_t(s,a_i)/T_m}}, \tag{20}$$

where $T_m > 0$ is called the *temperature* and controls how frequent the agent will choose random actions as opposed to the best actions $a_t^\star = \arg\max_a Q_t(s,a)$. If T_m decreases, the probability of generating the action with the highest expected reward $a_t^\star$ increases, leading to a decrease in the probability of selecting other actions and hence a lower probability of exploring the environment. In the limit of *zero temperature* $T_m \to 0$, the agent uses the target policy to select greedy actions in order to maximize the obtained rewards. At very large temperatures $T_m \to \infty$, all of the actions have almost the same probability and the action selection process approaches random walk. Setting the value for the temperature T is not straightforward, but it is generally reduced during the experiments, leading to more exploitation over time.

One of the extended applications of the *Softmax* exploration method is in the multiobjective reinforcement learning tasks (Vamplew et al., 2017), where scalar rewards are replaced with vector-valued rewards. Each element in the vectors represent the reward corresponding to an objective. The action-value function $Q(s,a)$ is consequently presented in the form of vector $\bar{Q}(s,a)$. In order to use the vector representation of the value functions in the Boltzmann formalism, the vectors must be mapped to scalar values using a scalarization function $f(\bar{Q}(s,a))$ (Liu et al., 2015), which can have either linear (Vamplew et al., 2017; Guo et al., 2009; Perez et al., 2009) or non-linear (Gábor et al., 1998; Van Moffaert et al., 2013a,b) representations. The Boltzmann formalism is consequently written as,

$$\pi\left(a|s\right) = \frac{e^{f\left(\bar{Q}_t(s,a)\right)/T}}{\sum_{i=1}^{N} e^{f\left(\bar{Q}_t(s,a_i)\right)/T}}. \tag{21}$$

Another extension of Softmax exploration is the *Max-Boltzmann* rule (Wiering, 1999), which is a combination of the ϵ-greedy or Max-random approach (explained in detail in section 4.1) and the Boltzmann exploration method. In the Max-Boltzmann method, similar to the ϵ-greedy approach, the action that gives the maximum Q-value is chosen with the probability $1-\epsilon$. With the probability ϵ, the Boltzmann distribution (equation 20) is used for

Approach	Remarks
Bridle (1990)	Softmax
Watkins (1989)	Early use of Boltzmann distribution
Lin (1992)	Early use of Boltzmann distribution
Barto et al. (1991)	Early use of Boltzmann distribution
Vamplew et al. (2017)	Vectorization of rewards in multi-objective tasks
Wiering (1999)	Max-Boltzmann (ϵ-greedy + Boltzmann)
Tokic (2010)	Value-Difference Based Exploration (VDBE)
Tokic and Palm (2011)	Extension of VDBE (VDBE + Max-Boltzmann)
Tijsma et al. (2016)	Controls and increases the probability of greedy action selection with time
Even-Dar and Mansour (2002)	Derandomization of ϵ-greedy

Table 2: Examples of value-based randomized action selection exploration methods.

action selection. A drawback for the Max-Boltzmann exploration method compared to the two approaches, Max-Random and Boltzmann, is the need for tuning two hyperparameters T and ϵ instead of one. However, the Max-Boltzmann method has been shown to reduce the weight of exploration in comparison with exploitation, and thus avoid over-exploration.

Another exploration technique in this category is the incremental Q-learning algorithm (Even-Dar and Mansour, 2002), which is an extension of the ϵ-greedy method (discussed in section 4). In particular, in their proposed method, Even-Dar and Mansour (2002) derandomize the ϵ-greedy method by adding a *promotion* term to the estimated Q-value for each state-action pair. If action a is not taken in state s at time t, the promotion term of that specific state-action pair at time $t+1$ is increased by a value called *promotion function*, and zeroed otherwise. The promotion function plays the role of a decreasing ϵ in the ϵ-greedy approach, and decreases with the number of times action $a' \neq a$ has been taken in state s. Consequently, the values of actions that have not been taken are promoted and the fraction of time the sub-optimal actions are chosen decreases with time and vanishes in the limit.

Another extension of the ϵ-greedy method is the Value-Difference Based Exploration (VDBE) (Tokic, 2010), where the ϵ parameter in the ϵ-greedy approach is replaced with a state-dependent exploration probability $\epsilon_t(s)$ instead of being hand-tuned. In VDBE, the initialization of the exploration probability $\epsilon_t(s)$ is done with $\epsilon_o(s) = 1$ for all states. At each time-step, the TD-error is computed, which serves as a measure for the agent's uncertainty. A larger TD-error for a state corresponds to a larger uncertainty, which consequently triggers a higher chance of exploration by assigning a larger value to the exploration rate $\epsilon_t(s)$ for that specific state.

Although Tokic (2010) assesses VDBE in multi-armed bandit problems, he argues that the method of VDBE is also applicable in multi-state MDPs. In another study, Tokic and Palm (2011) introduce an extension of VDBE, namely *VDBE-Softmax*, for solving reinforcement learning problems with multiple states. The VDBE-Softmax method combines

VDBE (introduced by Tokic (2010)) with Max-Boltzmann exploration (proposed by Wiering (1999)) in order to overcome the shortcomings of the two former approaches. A disadvantage of VDBE, as stated by Tokic and Palm (2011), is that it does not discriminate between exploratory actions, which leads to equal probability of selecting the actions that yield high and low Q-values. Another disadvantage of VDBE is its divergence in the cases where oscillations exist in the value functions caused by stochastic rewards or function approximators. In the VDBE-Softmax approach, with probability $\epsilon_t(s)$, the agent selects the exploratory actions using equation 20, and chooses the argmax of Q-value (greedy action) with the probability $1 - \epsilon_t(s)$. The authors show that in general, their proposed variation of VDBE performs better than the exploration methods ϵ-greedy, Softmax and pure VDBE in the environments with deterministic rewards (cliff-walking problem and bandit-world tasks) as well as the ones with stochastic rewards (bandit-world tasks).

Similar to the method of VDBE (Tokic, 2010), there exist other exploration strategies, which were originally proposed for and applied in the bandit problems, but were later performed in multi-state MDPs as well. One of these approaches is called *Pursuit* (Thathachar and Sastry, 1984), which maintains the probability of selecting greedy action as well as the value of different actions at each time step. As described in Sutton and Barto (1998b), in k-armed bandit problems, Pursuit algorithms initialize the probability of choosing an arm with $p_{t=0}(a) = 1/k$ for all actions $a = 1, \ldots, k$. At each time step t, the probability of selecting action a is calculated as

$$p_{t+1}(a) = \begin{cases} p_t(a) + \alpha\left(1 - p_t(a)\right) & \text{if } a = \arg\max_i Q_t(i) \\ p_t(a) + \alpha\left(0 - p_t(a)\right) & \text{Otherwise} \end{cases} \tag{22}$$

where $\alpha \in (0,1)$ is the learning rate. According to equation 22, the probability of selecting the greedy action increases with time, leading to fewer exploratory moves and more exploitation. The equivalence of equation 22 for the case of multi-state MDPs is given as follows (Tijsma et al., 2016)

$$\pi_{t+1}(s_t, a_t) = \begin{cases} \pi_t(s_t, a_t) + \alpha\left(1 - \pi_t(s_t, a_t)\right) & \text{if } a_t = a_t^\star \\ \pi_t(s_t, a_t) + \alpha\left(0 - \pi_t(s_t, a_t)\right) & \text{Otherwise} \end{cases} \tag{23}$$

where $a_t^\star = \arg\max_a Q(s_t, a)$. Tijsma et al. (2016) perform pursuit as well as other exploration techniques including Softmax and ϵ-greedy in random discrete stochastic mazes with one optimal goal and two sub-optimal goal states. Their results show that Softmax outperforms the other exploration strategies and that the ϵ-greedy method has the worst performance of all.

5.2 Policy-Search Based Methods

After having discussed methods that implicitly represent a policy using value function, we turn our attention to policy search methods. Policy search methods explicitly represent a policy, instead of, or in addition to, a value function. In the latter case, these methods are more specifically referred to as actor-critic methods. Most policy search methods learn a stochastic policy. The stochasticity in the policy is usually also the main driver of exploration in such methods. The different ways in which such perturbations can be applied will be the

focus of the next several sections[1]: after this short introduction, Section 5.2.1 will introduce the organizational principle for the section, with Sections 5.2.2 and 5.2.3 explaining the individual methods in detail.

Many policy search methods belong to the policy gradient family. These methods aim to update the policy in the direction of the gradient of the expected return ∇J^π. A basic way to do so is by calculating a finite-difference approximation of the gradient. In this approach, rollouts are performed for one or more perturbations of the original parameter vector, which are then used to estimate the gradient. When the system is non-deterministic, these estimates are extremely noisy, although better estimates can be obtained in simulation when the stochasticity of the environment is controlled by fixing the sequence of pseudo-random numbers (Ng and Jordan, 2000).

More sophisticated approaches are based on the log-ratio policy gradient (Williams, 1992). The log-ratio policy gradient relies on stochastic policies, and exploits the knowledge of the policy's score function (gradient of the log-likelihood). Stochastic policies for continuous actions based on the Gaussian distribution (Williams, 1992) are still frequently used (Deisenroth et al., 2013). For discrete action spaces, a Gibbs distributions with a learned energy function (Sutton et al., 2000) can be used instead.

The initialization of the exploration policy can be freely chosen. In some policy architectures, the amount of exploration is fixed to some constant or decreased according to a set schedule. In other architectures, the amount of exploration is controlled by learned parameters, possibly separate from other parameters (such as those controlling the 'mean action' for any given state). Policy search methods typically maximize the expected return, and thus probability mass tends to slowly be shifted towards a more greedy policy (usually resulting in a decreasing amount of exploration). These and more advanced strategies will be discussed in more detail in Sec. 5.2.4.

The discussed approaches differ in an important aspect: while in finite-difference methods (Ng and Jordan, 2000) the *parameters* of the policy are perturbed, the method proposed by Williams (1992) selects the *actions* stochastically. Where perturbations are applied at the level of parameters, they often affect an entire episode (*episode-based* perturbations). In contrast, classically action-space perturbations are often only applied for a single time step (*independent* perturbations).

In this section, we will focus on research in the area of policy search methods that introduce new exploration strategies or that explicitly evaluate the effects of different exploration strategies. We will focus on such policy search methods that are trained on a single task and where the policy has its own representation. Policies that are defined only in terms of value function are covered in Section 5.1. Policies explicitly optimizing over a distribution of tasks are covered with Bayesian and meta-learning approaches in Section 7. An overview of the methods we will cover in this section is given in Table 3. The table groups the method by type and coherence of perturbation that, like Deisenroth et al. (2013), we consider to be key characteristics of exploration strategies in policy search. The following subsection will give a more detailed explanation of these characteristics.

---

1. A more general overview of the properties of policy search methods is given in Deisenroth et al. (2013).

Approach	Perturbed space	Temporal coherence*	Remarks
Barto et al. (1983)	Action-space	Independent	-
Gullapalli (1990)	Action-space	Independent	-
Williams (1992)	Action-space	Independent	-
Morimoto and Doya (2001)	Action-space	Correlated	Multi-modal (hierarchy)
Nachum et al. (2019)	Action-space	Correlated	-
Wawrzynski (2015)	Action-space	Correlated	-
Lillicrap et al. (2016)	Action-space	Correlated	-
Haarnoja et al. (2017)	Action-space	Independent	Multi-modal
Xu et al. (2018)	Action-space	Independent	-
Kohl and Stone (2004)	Parameter-space	Episode-based	-
Sehnke et al. (2010)	Parameter-space	Episode-based	-
Rückstiess et al. (2010)	Parameter-space	Episode-based	-
"	Action-space	Episode-based	-
Theodorou et al. (2010)	Parameter-space	Episode-based	-
Stulp and Sigaud (2012)	Parameter-space	Episode-based	Correlated parameters
Salimans et al. (2017)	Parameter-space	Episode-based	-
Conti et al. (2018)	Parameter-space	Episode-based	-
van Hoof et al. (2017)	Parameter-space†	Correlated†	-
Plappert et al. (2018)	Parameter-space†	Episode-based†	-
Fortunato et al. (2018)	Parameter-space†	Episode-based†	-
Mahajan et al. (2019)	Parameter-space	Episode-based	Multi-agent

Table 3: Different exploration approaches proposed in the context of policy search algorithms. The first section of the table lists methods that mainly perturb the policy in the action space, these methods will be discussed in Sec. 5.2.2. The second section lists methods that mainly perturb the policy in the parameter space, that will be discussed in Sec. 5.2.3. Within these two broad categories, papers are ordered roughly chronologically, although papers within a similar line of work are kept together. Multiple entries for the same paper refer to different variants.
* Denotes whether perturbations are applied *independently* at each timestep, don't change at all throughout an *episode*, or have an intermediate *correlation* structure. Details in Sec. 5.2.1.
† These methods have additional step-based action-space noise for numeric reasons or to ensure a differentiable objective.

5.2.1 PERTURBED SPACE AND COHERENCE

In policy based methods, exploratory behavior is usually obtained by applying random perturbations. One of the main characteristics that differentiate exploration methods is where those perturbations are applied. Within policy gradient techniques, there are two main candidates: either the actions or the parameters are perturbed (although we will discuss some approaches beyond these two shortly). These possibilities reflect two views on the learning problem. On the one hand, the actions that are executed on the system are what actually affects the reward and the next state, no matter which parameter vector generated the actions. From this perspective, it is more straightforward to start with finding good actions, and subsequently find a parametric policy that can generate them. From another perspective, parametrized policies have limited capacity, and the resulting inductive bias might mean that the true optimal policy is excluded from the set of representable policies. We are then looking for parameters with which the overall policy behavior is best across all states. Also, if the structure of the policy is chosen to reflect prior information about the structure of the solution (e.g. policies linear in hand-picked features or policies with a hierarchical structure), perturbing the policy parameters ensure that explorative behavior follows the same structure.

The space in which explorative behaviors are applied is usually closely linked to the coherence of behavior. Coherence here refers to the question of whether (and how closely) perturbations in subsequent time steps depend on one another. Exploration with a low coherence (e.g., perturbations chosen independently at every time step) has the advantage that many different strategies might be tried within a single episode. On the other hand, exploration with a high coherence (e.g., perturbing the policy at the beginning of each episode only) has the advantage that the long-term effect of following a certain policy can be evaluated (Sehnke et al., 2010). Whereas independent perturbations can result in inefficient random walk behavior, following a perturbed policy consistently could result in reaching a greater variety of states (Machado et al., 2017). Intermediate strategies between the extremes of completely identical perturbation across an entire episode and completely independent perturbation per time step are also possible (Morimoto and Doya, 2001; Wawrzynski, 2015), and can be used to compromise between the advantages of the more extreme strategies.

Most exploration approaches which perturb the policy in action space have focused on independent perturbations in each time step, as applying the same perturbation at all time steps would not cover the space of possible policies well (see Table 3). In contrast, parameter-space exploration tends to go together with episode-based exploration, because certain parameters might only influence behavior in certain states, so such a perturbation has to be evaluated across multiple states to give a good indication of its merit (Rückstiess et al., 2010). However, exceptions to this pattern exist, especially in the area of exploration strategies of intermediate coherence. In the following paragraphs, papers presenting or analyzing specific exploration strategies will be discussed in more detail. We will start by discussing exploration strategies that apply explorative perturbations in the action space, before turning our attention to strategies that perturb the policy parameters. Finally, we will discuss the issue of what distribution these perturbations are sampled from.

5.2.2 Action-space perturbing strategies

Using stochastic policies to generate action-space perturbations has been used at least since the early 80's. Barto et al. (1983) proposed an early actor critic-type algorithm, that perturbed the output of a simple neural network before applying the thresholding activation function. This resulted in Bernouilli-distributed outputs (Williams, 1992).

Subsequent work also investigated the use of Gaussian noise in continuous action domains. Gullapalli (1990) introduced specific learning rules for the mean and standard deviation of Gaussian policies, which was introduced to learn policies with continuous actions. Williams (1992) provided a more general algorithm that provides an update rule for a general class of policy functions including stochastic and deterministic operations. Williams (1992) notes that because a Gaussian distribution has separate parameters controlling location and scale, such random units have the potential to control both the degree of exploration as well as where to explore. These early approaches all perturbed the action chosen at each time step independently.

While independent perturbation is still a popular method, attention has also turned to strategies that attempt to correlate behavior in subsequent time steps. An interesting early example can be found in Morimoto and Doya (2001). That paper describes a hierarchical policy, where an upper-level policy sets a sub-goal which a lower-level policy then tries to achieve. Exploration on the upper-level by itself causes some consistency in the exploration behavior, as a perturbation in the goal-picking strategy will consistently perturb the system's behavior until the subgoal is reached. Additionally, the lower-level learner itself applies a low-pass filter on the action perturbations. As a result, similar perturbations will typically be applied on subsequent time steps. The authors applied this algorithm to learn a stand-up behavior for a real robot.[2]

Nachum et al. (2019) studied hierachical methods in more detail, among others focusing on their exploration behavior. In their experiments, they investigate two simple exploration heuristics that share certain properties with hierarchical policies. The first heuristic, Explore & Exploit, randomly sets 'goals' for a separately trained explore policies analogous to the high-level actions in a goal-conditioned hierarchical method. Goals stay active for one or multiple time steps. Their second method, Switching Ensembles, trains several separate networks that individually attempt to optimize rewards. During training, the active policy is periodically switched, and when the policies are different, this switching leads again to exploration behavior that is coherent over several time steps. Nachum et al. find that both methods benefit from temporal coherence, and their results suggest that setting goals in a meaningful space might additionally benefit exploration.

Similar to the approach by Morimoto and Doya (2001), Wawrzynski (2015) investigated performing explorative perturbations on physical robots. As the authors note, applying perturbations independently at each time step (e.g., independent draws from a stochastic policy) causes jerkiness in the trajectories, which damages the robot. As an alternative, the paper proposes to apply an auto correlated noise signal. This signal is generated in a slightly different way than the previously discussed approach, as it is generated by summing

---

2. Other work has specifically evaluated the potential of lower-level sub-policies to decrease the diffusion time during the exploration process in the context of pure exploration (Machado et al., 2017). This paper is explained in more detail in Sec. 4.2.

up independent perturbations from the last M time steps. The authors explicitly evaluated the suggested strategy on various continuous robot control problems. Their experiments suggest that the proposed strategy leads to equivalent asymptotic performance (although sometimes a slower learning speed), while causing less stress to the robot's joints by reducing the jerkiness of trajectories.

Correlating the perturbations over several time steps, however, complicates the calculation of log-ratio policy gradients, as the policy is no longer Markov (as the selected action is no longer independent of earlier events given the state). Lillicrap et al. (2016) instead apply auto-correlated noise for their deep deterministic policy gradient (DDPG). Since DDPG is an off-policy algorithm, the generating distribution of the behavior policy does not need to be known, simplifying the use of various kinds of auto-correlated noise. They proposed generating this noise using an Ornstein-Uhlenbeck process, which generates noise with the same properties as that used by Morimoto and Doya (2001). This paper did not focus on real-robot experiments, thus motor strain was not a major concern. They did, however, determine that auto-correlated noise does help learn in (simulated) 'physical environments that have momentum', in other words, environments where a sequence of similar actions need to be performed to cause appreciable movement in high-inertia objects.

More recently, attention seems to have swung back to independent action perturbations, with recent work attempting to make the distribution from which actions are drawn more expressive, or the resulting explorative actions more informative. While classically, simple parametric distributions have been used as stochastic policies, these typically cannot represent multi-modal policies. Haarnoja et al. (2017) point out that it is useful to maintain probability mass on *all* good policies during the learning process, even if this results in a multi-modal distribution. In particular, a seemingly slightly-suboptimal mode might in a later stage of learning be discovered to actually be optimal, which would be hard to uncover if this mode was discarded earlier in the learning process. The authors define the exploration policy as maximizing an objective composed of a reward term and an entropy term. The solution to this maximization problem is an energy-based policy. As one cannot generally sample from such distributions, an approximating neural-network policy is fit to it instead. The authors show that with certain (initially) multi-modal reward distributions the method outperforms exploration using single-modal exploration policies. They also show empirically that a multi-modal policy learned on an initial task can provide a useful bias for exploring more refined tasks later.

Although maintaining a high entropy can be a useful strategy to obtain more informative data, it might be even more effective to directly maximize the amount of improvement to a target policy caused by data gathered using the exploratory behavior policy. This was the approach proposed by Xu et al. (2018), who study the optimization of the behavior policy in off-policy reinforcement learning methods, where the exploration policy can be fundamentally different from the target policy. The authors' insight is that good exploration policies might indeed be quite different from good target policies, and thus might not be centered on the current target policy, but instead have a separate parametrization. While the target policy is adapted in an off-policy manner in the direction of maximum reward, the behavior policy is separately updated by an on-policy algorithm towards greater improvements to the target policy. This is achieved by using an estimate of the improvement of the target policy as the reward of a 'meta-MDP'. Experiments show that learned variance, and even more

26

so a learned mean function, results in faster learning and better average rewards compared to conventional exploration strategies centered on the target policy. The authors attribute this performance gain to more diversity in the exploration samples leading to more global exploration.

5.2.3 Parameter-space perturbing strategies

Instead of using action-space perturbation, one might directly perturb the parameters of the policy. Especially where the parametrization of the policy can be restricted due to prior knowledge about the problem, it might be advantageous to do so. For example, if we know or assume that the optimal action will be directly proportional to the deviation from a set point, it is not informative to perturb the policy at the set point, because the policy will choose an action of 0 at the set-point regardless of the parameter value. This information is implicitly taken into account if parameters, rather than actions, are perturbed.

Perturbing parameters rather than actions also ensure that any explorative action can also, in fact, be reproduced by some policy in hypothesis space (Deisenroth et al., 2013). Referring back to the previous example, a non-zero action perturbation at the set-point, for example, would not be reproducible by any of the considered controllers.

Perturbing parameters also works very well together with temporally coherent exploration. A parameter vector might simply be perturbed only at the beginning of an episode, and then kept constant for the rest of the episode. Contrast this to action-perturbing schemes, where keeping the action perturbation constant for a whole episode in general would not yield behavior that covers the state-action space well.

The most straightforward way to find out what parameter perturbations improve a policy is to treat the entire interaction between the policy parameters and environment as a black-box system and calculate a finite-difference estimate of the policy gradient, keeping the policy perturbation fixed during the episode. An example of this approach is given by Kohl and Stone (2004). For each policy roll-out, each policy parameter randomly chosen to be either left as is or to be perturbed by a adding a small positive or negative constant. After obtaining a set of such roll-outs, the policy gradient is then estimated using a finite-difference calculation. This method was demonstrated to able to optimize walking behaviors on four-legged robots better than earlier hand-tuned or learned gaits.

A potential problem with this approach is that stochasticity (from the policy or environmental transitions) can make gradient estimates extremely high-variance. In simulation, where stochasticity can be controlled, this can be addressed by using common random numbers, as was proposed by Ng and Jordan (2000) in their PEGASUS algorithm to learn policies for gridworld problems and a bicycle simulator.

Ratio-likelihood policy gradient estimators exploit the knowledge of the parametric form of the policy to calculate more informed estimates of the policy gradient. Sehnke et al. (2010) proposed a stochastic policy gradient with parameter-based exploration by positing a parameterized distribution $\pi(\theta|\rho)$ over the parameters θ of a (deterministic) low-level policy, and learning the hyper-parameters ρ. Their experiments showed that the resulting parameter-perturbing, episode-based exploration strategy outperformed conventional action-perturbing strategies on several simulated dynamical systems tasks, including robotic locomotion and manipulation tasks. Rückstiess et al. (2010) extended the idea of parameter-

based exploration to several other policy-search algorithms, including a new method called state-dependent exploration. In that approach, perturbations are defined in the action-space, but are generated based on an 'exploration function' that is a deterministic function of a randomly generated vector and the current state. The authors show that state-dependent exploration is equal to parameter-based exploration in the special case of linear policies, and argue that in other cases it combines some of the advantages of parameter-based and action-based exploration. The resulting data was then used to perform REINFORCE updates. As these updates do not fully account for the dependency between actions, they might thus have an increased variance.

Theodorou et al. (2010) proposed 'per-basis' exploration, which is a variant parameter-based exploration scheme where the perturbation is only applied to the parameter corresponding to the basis function with the highest activation and kept constant as long as that basis function had the highest activation. Theodorou et al. (2010) noted that they emperically observed this trick improved the learning speed.

The effect of step-based (independent) versus correlated or episode-based exploration was further studied by Stulp and Sigaud (2012). They investigated connections between CMA-ES (Covariance matrix adaptation evolutionary strategies) from the stochastic search literature (Hansen and Ostermeier, 2001) and PI^2, a reinforcement learning algorithm with roots in the control community (Theodorou et al., 2010). Stulp and Sigaud (2012) found that episode-based exploration indeed outperformed per time-step exploration on a simulated reaching task. Furthermore, it also outperformed per-basis exploration proposed by Theodorou et al. (2010).

Salimans et al. (2017) applied the idea of episode-level log-ratio policy gradients (as used in earlier work by e.g. Sehnke et al. 2010) to complex neural network policies. They proposed the use of virtual batch normalization to increase the methods sensitivity to small initial difference. The authors connect this method to approaches from the evolutionary strategies community (e.g. Koutnik et al. 2010). Salimans et al. (2017) found their approach to compare favorably to Trust Region Policy Optimization (TRPO, Schulman et al. 2015), an action-space perturbing method, on simulated robotic tasks. Furthermore, the approach performed competitively with the Asynchronous Advantage Actor Critic (A3C, Mnih et al. 2016), while training an order of magnitude faster.

Conti et al. (2018) combined similar ideas from the evolutionary strategies community with directed exploration strategies. This combination results in two new hybrid approaches, where evolutionary strategies are used to maximize a scalarization of the original reward-maximization objective with a term encoding novelty and diversity. A heuristic strategy for adapting the scalarization constant is proposed. The proposed approach is evaluated on a simulated locomotion task and a benchmark of Atari games, where it outperforms a regular evolutionary strategy approach and performs competitively with the deep RL approach by Fortunato et al. (2018; described below). In this experiment, the evolutionary strategies were given more frames, but still ran faster due to better parallelizability.

Van Hoof et al. (2017) apply auto-correlated noise in parameter space. This noise is distributed similarly to that proposed by Morimoto and Doya (2001) and Lillicrap et al. (2016), but applied to the parameters rather than the actions. As a result, intermediate trade-offs between independent and episode-based perturbations are obtained. The latent parameter vector violates the independence assumption of step-based log-ratio policy gradients meth-

ods, which is resolved by explicitly using the joint log-probability of the entire sequence of actions. Expressing this log-probability in closed form requires the use of a restricted policy class (e.g., linear policies). Note that the auto-correlated action-space noise used by Lillicrap et al. (2016) was applied in an off-policy setting, avoiding this problem. Auto-correlated parameter-space noise was compared to several baselines, including action-space perturbations as well as episode-based and independent parameter-space perturbations. On various simulated and real continuous control tasks, intermediate trade-offs between independent and episode-based perturbation led to faster learning and a way to control state space coverage and motor strain.

Two methods (Plappert et al., 2018; Fortunato et al., 2018) independently proposed strategies to use parameter-based perturbations for reinforcement learning approaches based on deep neural networks. Both of these works consider both value-networks and policy-based approaches. Here, we will discuss the policy-based variants. Both of the approaches also build on the principle of episode-based perturbation of the parameter space, but better exploit the temporal structure of roll-outs than previous studies (Sehnke et al., 2010; Salimans et al., 2017) that largely ignored it.

By making the perturbations fixed over an entire episode and using the reparametrization trick, these methods allow the use of a wide range of policies, but possibly increase variance. Subsequent actions are now conditioned on a shared sample from the noise distribution, and their conditional independence means the trajectory likelihood can be factored as usual. As a result, this methods are applicable to non-linear neural network policies. Plappert et al. (2018) used a pre-determined amount of noise that was decreased over time according to a pre-set schedule. On the other hand, Fortunato et al. (2018) learned the magnitude of the parameter noise together with the policy mean

Applying this principle in an off-policy setting is relatively easy: since any behavior policy could be used to generate data, this could easily be the current deterministic target policy with noise added to the parameters. Plappert et al. (2018) modified the Deep Deterministic Policy Gradient (DDPG, Lillicrap et al. 2016) algorithm in this manner. With on-policy algorithms, this is more challenging. On-policy algorithms with per time-step updates tend to also require stochastic action selection in each time step. For the on-policy Asynchronous Advantage Actor Critic (A3C, Mnih et al. 2016) in Fortunato et al. (2018); and for Trust Region Policy Optimization (TRPO, Schulman et al. 2015) in Plappert et al. (2018); this problem was resolved by using a combination of parameter perturbations and stochastic action selection.

The NoisyNet-A3C method by Fortunato et al. (2018) compared favorably to the baseline A3C variant on a majority of 57 Atari games. The parameter-exploring variants of DDPG and TRPO proposed by Plappert et al. (2018) compared favorably to baselines with (correlated or uncorrelated) action-space noise on several simulated robot environment. [3]

Incoherent behavior causes particular problems in multi-agent learning, as pointed out by Mahajan et al. (2019). In cooperative decentralized execution scenarios, uncoordinated exploration between the agent can lead to state visitation frequencies for which the factorized

---

3. Colas et al. (2018), whose exploration method is discussed in Sec. 4, verify the performance difference between action- and parameter space noise for DDPG, and compare their methods to the exploration methods by Lillicrap et al. (2016) and Plappert et al. (2018). Like these methods, their proposal benefits from the flexibility of DDPG as off-policy method to work with data from any behavior policy.

q-function approximation is catastrophically bad. Such failure traps the agents in suboptimal behavior. Mahajan et al. (2019) remedy the situation by making exploration at train time coherent across both time and the individual agents, by conditioning on a common latent variable generated by a high-level policy[4]. A separate variational network is used to estimate a mutual information term which avoids collapse of the high-level policy on constant low-level behavior. The proposed approach is compared on both a toy task as well as challenging scenarios from the StarCraft Multi-Agent Challenge (Samvelyan et al., 2019).

5.2.4 THE DISTRIBUTION OF PERTURBATIONS

Separate from the issue of how perturbations are applied, is the issue of what distribution these perturbations are sampled from. Often, these are Gaussian distributions centered on the policy mean, leaving the choice of standard deviation open. Other parametric policies may have different parameters controlling the amount of exploration. Sometimes, such parameters are treated as additional hyperparameters (Silver et al., 2014) or governed by a specific heuristic (Gullapalli, 1990). More commonly, they can also be adapted like the other parameters during learning. The most straightforward way is to adapt the parameters controlling this standard deviation using the same policy gradient (Williams, 1992). Without additional regularization, policy gradient methods will tend to reduce the uncertainty, leading to a loss of exploration that is hard to control and might result in premature convergence to a suboptimal solution (Williams and Peng, 1991; Peters et al., 2010).

To address this problem, several approaches of them have been proposed. Many of them involve regularization using the entropy of the policy or the relative entropy from a reference policy. These entropies can either be constrained or added as regularization term to the optimization objective. A unified view on regularized MDPs is presented by Neu et al. (2017); Geist et al. (2019).

As an example, Bagnell and Schneider (2003) studied natural policy gradients through the lens of limiting the divergence between successive policies. They found the natural gradient can be derived from a bound on the approximate Kullback-Leibler divergence between trajectory distributions. Dynamic policy programming (Azar et al., 2011) uses a similar formulation but instead uses the relative entropy as penalty term. Schulman et al. (2015) provides a more exact method, focused on limiting the equivalent expected Kullback-Leibler divergence between policies. They connects this update to the idea of trust region optimization, and provides several steps to make scale this type of network to deep reinforcement learning architectures with tens of thousands of parameters.

The 'relative entropy policy search' method proposed by Peters et al. (2010) bounds the KL divergence in the joint state-action distribution to avoid a loss of exploration during training. Their bound on the joint KL is a stricter condition than a bound on the expected KL divergence of state-action conditionals which has theoretically attractive properties (Zimin and Neu, 2013). However, the method is more complex and seems harder to scale to deep architectures (Duan et al., 2016a). Bas-Serrano et al. (2020) propose a method building on relative entropy policy search, that combines regularization terms on the joint- and expected

---

4. Note that although the perturbed parameters are of a value network, this hybrid value- and policy based approach was covered here as the novelty in the exploration strategy stems from the high-level parametrized policy.

KL. A large benefit is that the resulting algorithm can be faithfully implemented in deep RL frameworks, and thus does need further approximations of the policy.

An alternative used early on was to add the derivative of the policy entropy to the policy updates. Williams and Peng (1991) found this strategy to improve exploration open on a toy example and several optimization problems. This strategy has also proved fruitful in practice in deep learning approaches: Mnih et al. (2016) applied this strategy and informally observed it lead to improved exploration by discouraging premature convergence. Such an entropy term can be seen as a special case of the expected relative entropy objective, with the reference policy being the maximum entropy distribution. Neu et al. (2017) studied such regularized objectives in detail, and conclude that a entropy penalty based on samples from the previous policy distribution distribution can lead to optimization problems.

The entropy regularization methods in the previous paragraph only took the instantaneous policy entropy into account. Haarnoja et al. (2017, 2018) instead propose two methods that explicitly encourages policies that reaching high-entropy states in the future. They note their method improves exploration by acquiring diverse behaviors.

6. Bonus-Based/Optimism-Based Exploration

A popular category of exploration methods commonly used in domains with weak or sparse explicit reward structure is the bonus-based methods. In this category, extrinsic reward $r(s, a)$ is augmented with a *bonus* term that is often demonstrated as a form of intrinsic reward (Oudeyer et al., 2007) to encourage better exploration. The term bonus was first introduced in the early nineties by Sutton (1990) in the tabular setting. Algorithms that adopt the bonus-based exploration approach employ different bonus calculation techniques to encourage the choice of action that leads to a higher level of uncertainty and consequently, novel or informative states.

In an environment with underlying MDP $\mathcal{M}$, upon selection of the state-action pair (s, a), the explicit reward $r(s, a)$ is observed by the RL agent, and the bonus term $\mathcal{B}(s, a)$ is computed by the RL agent. Thus, the total reward obtained by the agent by taking action a at state s is defined as,

$$r^+(\mathbf{s}, \mathbf{a}) := r(\mathbf{s}, \mathbf{a}) \oplus \mathcal{B}(\mathbf{s}, \mathbf{a}), \tag{24}$$

where the operator $\oplus$ denotes the aggregation between the two sources of environment (extrinsic) reward and bonus term.

In designing bonus-based exploration algorithms, two main questions arise:

1. How should the bonus function $\mathcal{B}(s, a)$ be specified to yield an effective exploration behavior?

2. How should we combine the two separately acquired sources of information, exploration bonus denoted by $\mathcal{B}(s, a)$ and *extrinsic* reward denoted by $r(s, a)$?

Exploration methods described in this section adopt three different approaches to compute the additive bonus term, namely *optimism-based*, *count-based*, and *prediction error-based*. In the optimism-based methods, the bonus term is implicitly embedded in the value function initial value estimates. In the count-based methods, where the novel state-action

pairs are the ones that are less frequently visited, the bonus term is calculated based on some form of state-action visitation counts. Another way of calculating the bonus term is through a prediction model of environment dynamics and measuring the induced prediction error. The interplay between the optimism-based approach and the count-based method in designing the bonus term is subtle, and it is often hard to distinguish between these two paradigms. Thus, in the optimism section, we only discuss the optimism-based approaches that do not explicitly use any notion of count in their choice of bonus. In the count-based section, we only discuss the methods that explicitly employ a notion of count in their bonus definitions. In this survey, we choose to interpret the optimistic initialization heuristic as a type of bonus assigned to unseen state-action pairs to encourage exploration.

All these three categories are divided into two sub-categories of *tabular* and *function approximation* methods. In the tabular setting, the state and action spaces are small enough that the value function estimate can be presented as a lookup table. In the function approximation setting, on the other hand, due to the infinite or large nature of state and action pairs, the value function is represented as a parameterized function rather than a table. At the beginning of each section, we provide a table that summarises the papers discussed.

6.1 Optimism-based methods

Optimistic exploration is a category of exploration methods that adopt the philosophy of Optimism in the Face of Uncertainty (OFU) principle, which was first introduced as an ad-hoc technique by Kaelbling et al. (1996); Kaelbling (1993); Sutton (1991a); Thrun and Möller (1992). From an optimistic perspective, a state is considered a good state if it induces a higher level of uncertainty in the state-value estimate and greater return potential. Optimistic exploration methods are typically realized by implicitly utilizing an *exploration bonus* either in the form of *optimistic initialization* (Even-Dar and Mansour, 2002; Sutton and Barto, 1998b; Szita and Lőrincz, 2008) or *Upper Confidence Bounds* (UCBs)(Strehl and Littman, 2008; Jaksch et al., 2010). In the optimistic initialization approach, the key assumption is that the unvisited state-action pairs yield the best outcome, whereas in the UCB-based methods, the unvisited state-action pairs are assumed to collect the outcome proportional to the largest statistically possible reward. In this section, we only focus on the methods that do not employ count-based approaches to implement the OFU principle as we address the count-based methods in a separate section.

6.1.1 Optimism-based methods: tabular

A model-based approach proposed as a generalization of E^3 algorithm (Kearns and Singh, 2002) (discussed in detail in Section 6.2.1) is the *R-max* algorithm (Brafman and Tennenholtz, 2002), which models agents' interactions in the context of zero-sum *stochastic games* (SG) instead of MDPs. In R-max (Brafman and Tennenholtz, 2002), the agent always maintains maximum likelihood estimates of environment dynamics and reward function if the observed data is sufficiently rich. The algorithm uses the approximate model estimates for a state-action pair if its visitation count exceeds a certain threshold. The optimistic approach is adopted at the initialization phase of the model, where all actions in all states are assumed to return maximum reward $R_{\max}$. R-max benefits from a built-in mechanism for resolving the exploration vs. exploitation dilemma because of the model estimation

Reference	Approach	Performance measure
	Tabular Methods	
Brafman and Tennenholtz (2002)	optimistic initialization	PAC bound
Auer and Ortner (2007)	UCB-based	PAC bound
Szita and Lőrincz (2008)	optimistic initialization	PAC bound/Empirical-Grid and Empirical-Toy MDP
	Function Approximation Methods	
Jong and Stone (2007)	optimistic initialization	Empirical-Grid world/Mountain Car
Nouri and Littman (2009)	optimistic initialization	PAC bound/Empirical
Ortner and Ryabko (2012)	UCB action selection	Regret bound
Kumaraswamy et al. (2018)	UCB action selection	Empirical-MuJoCo and Puddle World/ PAC Bound
Ciosek et al. (2019)	optimistic initialization	Empirical-MuJoCo
Rashid et al. (2020)	UCB action selection	Regret Bound/Empirical-Atari
Seyde et al. (2020)	UCB action selection	Empirical-MuJoCo

Table 4: Exploration methods that implement an optimism-based bonus mechanism.

and optimism. That is, taking the optimal action according to the learned model results in either exploring a previously unknown state or obtaining the near-optimal reward. In R-max, a state is marked as *known* if the number of states reachable (based on the learned model) from that state passes a fixed threshold; therefore, it is no longer considered a novel state. The sample complexity analysis conducted by Brafman and Tennenholtz (2002) shows near-optimal performance in a polynomial number of time-steps (assuming the state-space is finite). The R-max algorithm also attains near-optimal polynomial time average reward in $|\mathcal{S}|, |\mathcal{A}|$ and mixing time T.

In a similar approach, authors in Szita and Lőrincz (2008) propose a new sample efficient and model-based exploration algorithm called Optimistic Initial Model (OIM) in an MDP framework with finite state and action spaces. The proposed method assigns an optimistic value to unknown areas, and if the sampled state is among the set of 'Garden of Eden' states, it receives the maximum reward of $R_{\max}$. The RL agent in this method builds an approximate model of environment dynamics and updates the value functions using dynamic programming. To handle the full update sweep complexity imposed by dynamic programming, authors adopt the prioritized sweeping algorithm of Wiering and Schmidhuber (1998). Theoretically, the authors show almost sure convergence of the proposed method to a near-optimal policy in polynomial time under a lower-bound assumption on $R_{\max}$. Experimentally, OMI's performance is assessed against ϵ-greedy, Boltzmann and some other exploration methods in three environments of *RiverSwim* and *SixArms* (Strehl et al., 2006), *Maze* (Wiering and Schmidhuber, 1998) and *Chain*, *Loop* and *FlagMaze* (Meuleau and Bourgine, 1999; Strens, 2000; Dearden et al., 1999).

The seminal idea of employing optimistic *upper confidence bound* (UCB) to encourage optimistic exploration policies in RL setting was first introduced by Strehl and Littman (2004). Auer and Ortner (2007) proposed UCRL algorithm as the first near-optimal exploration method that extends the idea of optimistic UCBs in RL. UCRL computes a count-based upper bound on the empirical estimates of reward and transition probability after each visit and then switches between policies based on the observed gap and calculates and a new policy. Later, Jaksch et al. (2010) proposed UCRL2 as an extension to Auer and Ortner (2007). Apart from optimistic initialization, both UCRL and URCL2 implement count-based UCBs to encourage exploration. Thus, we postpone further explanation regarding these methods to the count-based section.

6.1.2 OPTIMISM-BASED METHODS: FUNCTION APPROXIMATION

After the successful application of OFU to RL with finite state-action MDPs, which we addressed in Section 6.1.1, some recent approaches extended this idea to MDPs with large or infinite state-action spaces (Azar et al., 2017; Bartlett and Tewari, 2012; Fruit et al., 2018; Filippi et al., 2010; Jaksch et al., 2010; Tossou et al., 2019). This section provides a comprehensive overview of the methods that employ OFU in the MDPs with large or infinite state-action spaces.

The study presented by Jong and Stone (2007) is a model-based exploration-exploitation trade-off algorithm for continuous state spaces, termed Fitted R-max. The proposed algorithm is a combination of R-max (Brafman and Tennenholtz, 2002) with fitted value iteration (Gordon, 1995). The algorithm first updates environment models at each time-step and then applies the value iteration step to solve their proposed Bellman optimality equation. In discrete setting, the proposed method simply implements the optimistic value function proposed by Brafman and Tennenholtz (2002) and control the exploration-exploitation trade-off through a visitation count threshold. In continuous setting, the authors propose a new counting method based on the sum of the unnormalized kernel values based in the estimated environment dynamics. The performance of the Fitted R-max algorithm is experimentally tested in the two environments Mountain Car (Sutton and Barto, 1998b) and Puddle World (Kearns and Singh, 2002).

Another line of work that implements the OFU principle in continuous state space environments is Nouri and Littman (2009), where the authors combine their proposed Multi-resolution Exploration (MRE) algorithm with fitted Q-iteration (Antos et al., 2008). Their proposed model-based method is built upon Kakade et al. (2003) and introduces a method that measures the uncertainty associated with the visited states through building regression trees, termed as *knownness-tree*. Knownness-tree is used to model the environment transition dynamics and optimistically update its model at each time step. Theoretically, Nouri and Littman (2009) show, under some smoothness assumption on transition dynamics, the near-optimal performance of the proposed algorithm, and assess the performance of MRE against ϵ-greedy algorithm empirically in continuous Mountain Car environment (Sutton and Barto, 1998a).

Kumaraswamy et al. (2018) propose a model-free computationally efficient exploration strategy based upon computing Upper-Confidence Least-Squares (UCLS), which are UCBs for least-squares temporal difference learning (LSTD). Since LSTD maintains the agent's

past interactions efficiently, the computed upper confidence bounds induce context-dependent variance, which encourages the exploration of states with higher variance. This study provides the first theoretical results that obtain UCBs for policy evaluation using function approximation. Empirically, UCLS algorithm shows outperformance over DGPQ (Grande et al., 2014), UCBootstrap (White and White, 2010), and RLSVI (Osband et al., 2016b) in Sparse Mountain Car, Puddle World and RiverSwim environments.

Ciosek et al. (2019) provide an optimistic actor-critic algorithm to resolve two phenomena: *pessimistic under-exploration*, that is deviating the algorithm from sampling actions that result in improvement on the critic estimates, and *directionally uninformed action sampling*, which is uniform sampling of actions that are lying in two opposite sides of mean in Gaussian policies. They state that these two phenomena prevent state-of-the-art actor-critic-based algorithms such as SAC (Haarnoja et al., 2018) from performing efficient exploration. To tackle these challenges, they calculate approximate upper confidence bounds on the value function estimates to encourage directed exploration and lower confidence bound to prevent overestimation of the value function estimates. They benchmark their proposed algorithm (OAC) in high-dimensional MuJoCo tasks, and the plotted results demonstrate marginal improvement against the SAC algorithm.

To avoid the pessimistic initialization phenomenon, commonly used in deep network initialization schemes, Rashid et al. (2020) propose an optimistic initialization algorithm, termed Optimistic Pessimistically Initialised Q-Learning (OPIQ) that decouples optimistic initialization of Q function from network initialization. Rashid et al. (2020) propose a simple count-based bonus augmented with the Q-value estimates. In the tabular setting, their proposed algorithm is based on Jin et al. (2018). In the Deep RL setting, they adopt commonly employed methods of calculating pseudo counts such as Bellemare et al. (2016); Ostrovski et al. (2017) to compute the additive bonus term. OPIQ is evaluated in the three domains toy randomized Markov chain, Maze and Montezuma's Revenge against the naive extension of UCB-H (Jin et al., 2018) to the deep RL and some variations of DQN augmented with some state-of-the-art pseudo-count estimate methods.

When both the environment dynamics and task objective are unknown to the RL agent, Seyde et al. (2020) propose a model-based exploration algorithm, termed Deep Optimistic Value Exploration (DOVE), to encourage deep exploration through adopting optimistic value function. Throughout each episode, DOVE learns a transition function and reward function using supervised learning. The initial conditions are applied to the learning policy generated by perturbing locally observed states fetched from the replay memory. The local perturbation performed throughout each episode is employed to ensure information propagation. Empirically, Seyde et al. (2020) benchmark DOVE in some high-dimensional continuous control MujuCo tasks.

6.2 Count-based bonus

One way to model the intrinsic reward is to measure how surprising a state-action pair is. An intuitive approach to measuring surprise is to count how frequently a particular state-action pair is visited. In the count-based setting, the notion of bonus is defined as a function of state-action pair visitation count. In table 5, we provide a list of exploration methods that

Reference	Bonus Type	Performance measure
Tabular Methods		
Sutton (1991b)	count-based	Empirical-Grid world
Moore and Atkeson (1993)	count-based threshold/optimistic initialization	Empirical-Grid world
Kaelbling (1993)	count-based	Empirical-Toy MDP
Dayan and Sejnowski (1996)	count-based	Empirical-Grid world
Tadepalli and Ok (1998)	count-based/optimistic initialization	Empirical-Grid world/AGV-scheduling
Kearns and Singh (2002)	count-based/optimistic initialization	PAC bound
Kakade et al. (2003)	distance-based count	PAC bound
Strehl et al. (2006)	UCB-based	PAC bound
Auer and Ortner (2007)	UCB-based	PAC bound
Jaksch et al. (2010)	UCB-based	PAC/Regret bound
Azar et al. (2017)	UCB-based	Regret bound
Kolter and Ng (2009)	count-based	PAC bound
Pazis and Parr (2013)	optimistic initialization / distance-based bonus	PAC bound/Empirical
Guo and Brunskill (2015)	count-based threshold	PAC bound
Jin et al. (2018)	count-based/UCB-based	PAC bound
Wang et al. (2020)	count-based/UCB-based	PAC bound
Function Approximation Methods		
Bellemare et al. (2016)	density-based	Empirical
Fu et al. (2017)	density-based	Empirical
Martin et al. (2017)	density-based	Empirical
Tang et al. (2017)	count-based	Empirical
Machado et al. (2018a)	count-based	Empirical
Rashid et al. (2020)	optimistic initialization/ count-based	Empirical

Table 5: Count-based methods.

explicitly employ a form of visitation count to control the exploration-exploitation trade-off.

6.2.1 Count-based bonus: tabular

In the tabular setting, counting visited states or state-action pairs is a trivial problem and the bonus term is typically used in one of the following forms,

$$\mathcal{B}(\mathbf{s}, \mathbf{a}) \text{ or } \mathcal{B}(\mathbf{s}) \propto \begin{cases} \sqrt{\frac{\ln n}{N(\mathbf{s},\mathbf{a})}} \text{ or } \sqrt{\frac{\ln n}{N(\mathbf{s})}}, \\ \frac{1}{\sqrt{N(\mathbf{s},\mathbf{a})}} \text{ or } \frac{1}{\sqrt{N(\mathbf{s})}}, \\ \frac{1}{N(\mathbf{s},\mathbf{a})} \text{ or } \frac{1}{N(\mathbf{s})}. \end{cases} \tag{25}$$

where n denotes the total number of time-steps taken by the RL agent.

An intuitive interpretation of equation (25) is that the bonus for visiting a state-action pair (s, a) is highest when (s, a) is novel, and decays each time the pair (s, a) is revisited. The main limitation of such definitions of bonus is that they are mainly applicable in tabular settings, where the set of state-action pairs is countable and finite. Although bonus-based methods employed in the tabular settings are not necessarily suitable for large state-and-action space settings, they still provide useful intuitions. The first study that employed the notion of bonus in the context of exploration algorithms was Sutton (1991b), where a new RL architecture Dyna-Q+ was proposed. Dyna-Q+, which is a combination of Dyna architecture (Sutton, 1991a) and Watkins Q-learning (Watkins and Dayan, 1992) uses an additional explicit count-based exploration bonus assigned to state-action pair (s, a). The bonus term used in Dyna-Q+ is proportional to the square root of the number of time steps that have elapsed after the last trial of action a at state s. This exploration bonus is added to the update rule designed to update Q-value. The main advantage of using such bonuses is to increase the chance of visiting the state-action pairs that have not been frequently visited. Sutton (1991a) tests his proposed model in the two RL environments Blocking and Shortcut tasks, designed as a small $2D$ maze environment against two other variations of Dyna-Q that do not use exploration bonus to encourage exploration. Sutton (1991a) shows that in both experiments, Dyna-Q+ outperforms other variations of Dyna-Q in the setting, where the performance is measured with respect to the collected reward. To address the two issues, namely high computational expense raised by Kaelbling (1993), and instability of the bonus raised by Sutton (1991a) in large state and action spaces, Moore and Atkeson (1993) proposed the *prioritized sweeping* algorithm that uses a preset threshold parameter T_{board} to determine whether the state-action pair is worth exploring more or not. Prior to reaching the visitation threshold, the bonus parameter is set to the max return value in the discounted reward setting $R_{\max}/(1 - \gamma)$. Once the visitation count exceeds the optimistic threshold, the algorithm uses the non-optimistic true discounted return. Apart from the tabular nature of this approach, its main bottleneck is that the key hyperparameter T_{board} is prefixed and set manually. The proposed algorithm's performance is experimentally tested against Dyna-Q in two deterministic and stochastic grid world environments.

An early study that adopts the OFU principle in a model-based setting for exploration is Tadepalli and Ok (1998). The proposed *H-learning* algorithm is designed in the context of the *average-reward RL* setting, where the RL agent's goal at each time-step t is to optimize the average expected reward. At each time-step t, an empirical model of the environment is computed, and consequently, a set of greedy actions accessible from the current state with respect to the Bellman equation for average-reward RL is reported, as well as the expected

long-term advantage function $h(s)$. The long-term advantage function $h(s)$ reflects the long-term impact of start state s on the obtained expected average reward. Eventually, the final expected average reward associated with actions in the set of greedy actions at the current state s is calculated with respect to $h(s)$, a temperature parameter α and expected average reward computed at time $t - 1$. Experimentally, the proposed H-learning algorithm is compared with four other exploration methods, random exploration, counter-based exploration, Boltzmann exploration, and recency-based exploration in a two-dimensional grid world with discrete state and action spaces, termed Delivery domain. The non-tabular version of the H-learning algorithm is proposed based on local linear regression as the function approximator and Bayesian network representation of the action space. The extended H-learning method called *LBH-learning* is tested in three AGV-scheduling tasks (Maxwell and Muckstadt, 1982) and compared to six different H-learning baselines.

The *Explicit Explore or Exploit* algorithm (known as E^3) (Kearns and Singh, 2002) adopts a model-based approach that initiates the exploration phase by dividing the set of states into two categories of known and unknown states. A state is considered to be known if the number of state visitations passes a certain threshold such that the learned dynamics are sufficiently close to the true one. If the current state is unknown, the algorithm calls the procedure of *balanced wandering*, in which the algorithm chooses the least frequent action at the unknown state and assigns the max reward to the unknown state. When the algorithm is not engaged in the balanced wandering phase, it performs two offline optimal policy computations sub-routines. Later, Kakade et al. (2003) proposed Metric-E^3 algorithm as a generalization of $E^3 algorithm$. Metric-E^3 provides the time complexity bound on finding a near-optimal policy that depends on the covering numbers of the state space rather than the size of the state space as presented by Kearns and Singh (2002). This difference is mainly due to the difference in their definition of a "known" state.

In the context of undiscounted RL, Auer and Ortner (2007) use the notion of upper confidence bounds to manage exploration-exploitation trade-off. In their study, count-based confidence bound proportional to $\sqrt{\frac{1}{N(s,a)}}$ is updated at each step and, together with empirical estimates of reward and transition functions, help the agent to control the exploration-exploitation trade-off. The regret analysis performed by Auer and Ortner (2007) shows logarithmic performance in the number of time steps taken by the algorithm based on the optimal policy.

Kolter and Ng (2009) provide an explicit notion of bonus, called *Bayesian Exploration Bonus*, to manage exploration-exploitation trade-off. Their proposed algorithm focuses on the Bayes-adaptive RL setting with a tabular representation of state-action space. The bonus term is proportional to $\frac{1}{1+N(s,a)}$, where $N(s, a)$ is calculated based on the number of visitation counts implied by the prior. Kolter and Ng (2009) provide a template for count-based bonus terms in the form of a theorem stating that any algorithm that adopts an exploration bonus of the form $\frac{1}{(N(s,a))^p}$ with $p \leq 1/2$ is not a PAC-MDP. In their proposed algorithm, called BEB, the action-selection is performed with respect to the mean of the current learned belief over transition model, with an additional Bayesian bonus. In the main theorem of this work, the authors show that their proposed algorithm, while allowing a higher rate of exploration, provides a near-optimal sample complexity bound, which is

polynomial in $|\mathcal{S}|$, $|\mathcal{A}|$, and time horizon T, where the optimality is defined in the Bayesian sense.

In the continuous state space setting, Pazis and Parr (2013) introduce C-PACE as a PAC-optimal exploration algorithm for continuous state MDPs. C-PACE adopts the OFU principle in the estimation of the Bellman equation. At each time step, from the k-nearest neighbours, the action that maximizes the optimistic Q value function is selected. The optimistic Q value function is defined based on the knowledge of k-neighbouring state-action pairs (the bonus term) and the immediate reward obtained upon transiting to any neighbouring pairs. C-PACE assumes the existence of a Lipschitz continuous distance metric over the set of state-action pairs. The main result of this paper provides a PAC bound that shows the near-optimal C-PACE performance with respect to the covering number of the state-action space. Finally, the authors evaluate the performance of C-PACE in a simulated HIV treatment environment.

Guo and Brunskill (2015) propose a confidence-based exploration algorithm called PAC-EXPLORE in a model-based setting, which is operationally very similar to the E^3 algorithm (Kearns and Singh, 2002), with the difference in the confidence bounds used to compute policies that are practically more efficient. The PAC-EXPLORE algorithm takes a state-action pair visitation threshold and divides the space of state-action pairs into two clusters of known and unknown pairs. If the state falls into the set of known states, the algorithm applies the same technique as in Wiering (1999) to estimate confidence bounds on transition probability. Authors in this work show that by sharing the experience of concurrently running agents on top of Wiering (1999), one can achieve linear improvement on the algorithm's sample complexity.

Delayed Q-learning (Strehl et al., 2006) is one of the first papers that study model-free PAC optimal algorithm. At each time-step t, the agent keeps track of three values for each visited state-action pairs (s, a), the value function $Q_t(s, a)$, the Boolean flag $\text{LEARN}_t(s, a)$ that indicates whether or not the change has occurred to the Q estimate for (s, a), and a visitation counter $N(s, a)$. The exploration-exploitation trade-off is managed based on a visitation count threshold and the value $\text{LEARN}_t(s, a)$. When the visitation count for (s, a) is larger than a pre-set threshold and the $\text{LEARN}_t(s, a)$ is true, the $Q_t(s, a)$ estimate is updated. At the initial phase, the Boolean flag $\text{LEARN}(s, a)$ is set to TRUE for all state-action pairs and $N(s, a)$ is set to zero. Strehl et al. (2006) under certain assumptions prove that their proposed algorithm is PAC-MDP in the tabular setting.

Jin et al. (2018) provide two types of upper confidence-based bonuses for Q-learning in the episodic tabular MDP setting: 1) Hoeffding-style bonus, 2) Bernstein-style bonus. By employing the Hoeffding-style bonus, the authors show $\mathcal{O}(\sqrt{T})$ regret dependency with respect to the total number of time-step T. They also show $\sqrt{H}$ improvement by using Bernstein-style bonus over the Hoeffding-style bonus exploration algorithm, where T denotes the time horizon.

Wang et al. (2020) introduce another method that addresses the sample efficiency of model-free algorithms by adopting UCB-exploration bonus in Q-learning. Their proposed UCB-based algorithm maintains two types of visitation counts, $N_t(s, a)$ that denotes the number of times the pair (s, a) has been visited up to time-step t, and $\tau(s, a, k)$ that records the number of time steps that state-action pair (s, a) has occurred for the k-th time. If (s, a) has not been visited k times, then $\tau(s, a, k) = \infty$. At each time-step, a bonus term

proportional to $\sqrt{\frac{|\mathcal{S}||\mathcal{A}|\ln(N_t(s,a))}{N_t(s,a)}}$ is added to the discounted value estimate, and the action-value function gets updated accordingly. To assess the PAC efficiency of the proposed algorithm, the authors first propose a learning instance illustrating $\Omega(1/\epsilon^3)$ lower bound incurred by Delayed Q-learning, which leaves a quadratic gap in $1/\epsilon$ from the best known lower bound in the class of UCB-based exploration algorithms.

6.2.2 COUNT-BASED: FUNCTION APPROXIMATION

Despite the near-optimal performance guarantees often achieved in the tabular setting, these methods are mostly not suitable for environments with large or infinite state spaces. This section summarizes exploration methods that adopt a notion of visitation count to design exploration algorithms for environments with large state and action spaces.

Bellemare et al. (2016) revisit the problem of extending count-based exploration to non-tabular setting and propose a density model that hinges upon ideas from the intrinsic motivation literature (refer to section 4) and propose an algorithm that measures state novelty for any choice of action given an arbitrary density model. The key contribution of their study is drawing a connection, called *pseudo-count*, between intrinsic motivation and count-based exploration. Pseudo-count quantity is derived from an arbitrary density model over the state space. The density model proposed in Bellemare et al. (2016) models a marginal distribution in which states are independently distributed. For any given choice of density model ρ, the paper draws a connection between two unknowns: 1) pseudo-count function and 2) pseudo-count total. The paper also introduces a connection between the conditional probability assigned to state s using ρ after observing its new occurrence conditioning based on its prior observations, pseudo-count function and pseudo-count total. The notion of information gain as a popular measure of novelty and curiosity is then shown to be related to pseudo-count, which leads to the main theorem in the paper that suggests using pseudo-count bonus leads to an exploratory behaviour as good as when the information gain bonus is used. Under two major assumptions: 1) the given density model asymptotically behaves similarly to the limiting empirical distribution, and 2) the learning rate at which ρ changes with respect to the true state distribution μ is positive in the asymptotic sense, the limit of ratios of pseudo-counts to empirical counts is finite and exists for all states. Bellemare et al. (2016) test their proposed method in comparison with two state-of-the-art RL algorithms, Double DQN (van Hasselt et al., 2016) and A3C (Asynchronous Advantage Actor-Critic) (Mnih et al., 2016) on some of the Atari 2600 games.

In a subsequent work, Ostrovski et al. (2017) answer two questions regarding the modeling assumptions raised in Bellemare et al. (2016): 1) what is the impact of the quality of density model on exploration? 2) To what extent do Monte-Carlo updates influence exploration? To address the first question, Ostrovski et al. (2017) adopt an advanced neural density model PixelCNN (Van den Oord et al., 2016), and discuss the challenges involved in this approach in terms of model choice, model training and model use. PixelCNN is a convolutional neural network that models a probability distribution over pixels conditioned on the previous pixels. The paper provides a list of properties that the density model requires and subsequently suggests a suitable notion of pseudo-count for DQN agents that leads to state-of-the-art results in difficult Atari games like Montezuma's Revenge.

Following the pseudo-count technique proposed by Bellemare et al. (2016) and Ostrovski et al. (2017), Martin et al. (2017) propose a new density model to measure the similarity between states and, consequently, a generalized visitation count method. Even though Bellemare et al. (2016) construct the density model over raw state visitations, the method proposed by Martin et al. (2017) relies on the feature map used in value function approximation to construct the density model. The bonus-based exploration algorithm ϕ-*Exploration Bonus* proposed by Martin et al. (2017) augments the extrinsic reward with the bonus term proportional to the inverse of the square root of the pseudo-count calculated based on the proposed feature-based density model. Empirically, ϕ-*Exploration Bonus* algorithm is evaluated against the ϵ-greedy, A3C (Mnih et al., 2016), Double DQN (Hasselt et al., 2016), Double DQN with pseudo-count (Bellemare et al., 2016), TRPO (Schulman et al., 2015), Gorila (Nair et al., 2015), and Dueling Network (Wang et al., 2016b) baselines in five different games from the Arcade Learning Environment (ALE).

Fu et al. (2017) introduce another study that uses count-based bonuses to conduct exploration in high-dimensional domains using the notions of curiosity and novelty. Effective exploration methods that are based on a notion of visitation novelty typically require either a tabular representation of states and actions or a generative model over state and actions, which can be difficult to train in high-dimensional and continuous settings. Fu et al. (2017) propose an approach to approximate state visitation densities using a discriminative model (exemplar model) over the complex model of states using deep neural networks, where the classifier assigns reward bonuses if the recently visited state is novel. The authors of this work show that discriminative modeling is equivalent to implicit density estimation. They argue that learning a discriminative model using standard convolutional classifier networks in the case of rich sensory inputs like images is typically easier than learning the generative model of the environment. Their proposed model is inspired by the concept of Generative Adversarial Networks (Goodfellow et al., 2014) and employs the intuition that novel states are typically more easily distinguished from all other visited states. The main idea is to maintain a density estimator using exemplar models based on a discriminatively trained classifier instead of maintaining explicit counts. To train the proposed discriminator, a cross-entropy loss is employed. Fu et al. (2017) evaluate their proposed method in sparse-reward continuous high-dimensional control tasks in MuJoCo (Todorov et al., 2012) and Vizdoom (Kempka et al., 2016). They compare the algorithm's performance with the two state-of-the-art baseline by Houthooft et al. (2016) (discussed in Section 6.3.2) and Schulman et al. (2015).

As an extension of count-based exploration to high-dimensional and continuous deep RL benchmarks, Tang et al. (2017) use a hashing mechanism to map novel states and visited states to hash codes and subsequently count the state visitations using the corresponding hash table. In the simple domains, authors propose a static hashing approach, in which the state space is discretized using a hash function such as SimHash (Charikar, 2002), and subsequently the bonus term is set to be proportional to the inverse of the square root of state count with respect to the hash code. In environments with complex structures, the authors adopt the Learned Hashing mechanism that implements an autoencoder to learn the appropriate hash codes. Like the static hash mechanism, the Learned Hashing mechanism also employs a bonus term proportional to the inverse of the square root of count on the

hash codes. This approach outperforms the method presented by Houthooft et al. (2016) in some rllab benchmark tasks, as well as the vanilla DQN agent in some Atari 2600 games.

Inspired by the results from Wu et al. (2018) and Machado et al. (2017), authors of Machado et al. (2020) show that the inverse of l_1 norm of successor representation (Dayan, 1993) can be interpreted as an exploration bonus in both tabular and function approximation setting. Successor representation (Dayan, 1993) can be interpreted as an implicit estimator of the transition dynamics of the environment. In the tabular setting, they augment the Sarsa (Sutton, 1992) update rule with the inverse of the norm of the successor representation of the visited states. Their proposed algorithm is empirically compared with traditional Sarsa in traditional PAC-MDP domains SixArms and RiverSwim (Strehl and Littman, 2008). In the function approximation case, the bonus used is similar to the one utilized in tabular setting and is the inverse of l_1 norm of the parameterized successor feature vector. Machado et al. (2020) evaluate their proposed algorithm empirically in the Arcade Learning Environments (Bellemare et al., 2013) with sparse reward structure, including Freeway, Gravitar, Montezuma's Revenge, Private Eye, Solaris, and Venture.

6.3 Prediction error-based bonus

In this category of exploration methods, the bonus term is computed based on the change in the agent's knowledge about the environment dynamics. The agent's knowledge about the environment is often measured through a prediction model of environment dynamics. This section focuses on the exploration techniques that use Prediction Error (PE) as an exploration bonus. PE is a term used to measure the difference between the true environment model parameters and their estimates that are used to predict transition dynamics. The methods that fall into this category use the discrepancy between the induced prediction from the learned model of environment dynamics and state-action representation models, and the real observation to assess the novelty of the visited states. The states that lead to more considerable discrepancy are considered more informative than those with a smaller discrepancy. The first two studies that employ PE as an exploration bonus to encourage curiosity are Schmidhuber (1991a) and Schmidhuber (1991b), which are explained in detail in Section 4 (due to the fact that they can also function in environments that do not provide extrinsic rewards).

Formally, let H_t be the history of observations until time-step t, a_t denote the action taken at time t, and M_ϕ be the predictive model of transition parameterized by the feature function ϕ. Then, the prediction error at time t is proportional to,

$$e(H_{t-1}, a_t, s_t) \propto \|s_t - M_\phi(H_{t-1}, a_t)\|_p, \tag{26}$$

where $\|.\|_p$ denotes the p-norm of a given vector.

6.3.1 Prediction error-based bonus: Tabular

Dayan and Sejnowski (1996) consider the problem of exploration in a non-stationary absorbing finite POMDP setting and provide a systematic approach to designing exploration bonuses in such setting. Their algorithm borrows the certainty equivalence approximation technique from the dual control literature and provides a statistical model of the environment's uncertainty in a finite state space setting and subsequently incorporates the uncer-

Reference	Approach	Performance measure
	Tabular Methods	
Schmidhuber (1991a)	Confidence-based	Empirical-Grid World
Dearden et al. (1998)	Information gain-based	Convergence/Empirical
Wiering (1999)	Confidence-based	PAC bound
Ishii et al. (2002)	Confidence-based/Entropy-based	Empirical-Grid
Strehl and Littman (2004)	Confidence-based	Empirical
Strehl and Littman (2008)	Confidence-based	PAC bound/Empirical
Lopes et al. (2012)	density-based	PAC bound/Empirical
	Function Approximation Methods	
White and White (2010)	Confidence based	Convergence/Empirical
Stadie et al. (2015)	PE-based bonus	Empirical
Houthooft et al. (2016)	Information gain	Empirical
Pathak et al. (2017)	PE-based bonus	Empirical
Burda et al. (2018a)	PE-based bonus	Empirical
Hong et al. (2018)	PE-based bonus	Empirical
Burda et al. (2018b)	density-based	Empirical
Kim et al. (2019)	Information gain-based	Empirical

Table 6: Prediction Error-based methods

tainty estimates into the systematic design of exploration bonuses. The exploration bonus in Dayan and Sejnowski (1996) is proportional to the amount of uncertainty induced by the agent's belief system and the true model of the environment. Dayan and Sejnowski (1996) assess their proposed method against DAYNA (Sutton, 1991a) in a two-dimensional maze with movable barriers.

The concept of Interval Estimation (IE) was first introduced by Kaelbling (1993) in the bandit setting to employ confidence intervals during the exploration phase. The agent chooses the action that induces the highest upper confidence bound. A few years later, Wiering (1999) provided a theoretical extension to Kaelbling (1993) and discussed a new variation of Model-Based Interval Estimation (MBIE) by augmenting the Bellman optimally equation with a bonus term proportional to $1/\sqrt{N(s,a)}$. The author also provide a formal PAC-style guarantee for their proposed algorithm and theoretically analyze the effect of additive bonus term on the number of time steps required to achieve the sub-optimal performance.

In the stream of model-based approaches to exploration, Ishii et al. (2002) compute the exploration bonus using the entropy of the posterior distribution of the state-transition kernel. The reward associated with the state-action pair is composed of the obtained immediate reward, and the entropy of the visited state-action pairs. The action sampling policy is based on a softmax action selection algorithm combined with an entropic bonus term. Small entropy means that the amount of information acquired given the agent's current model of the environment is expected to be small, and therefore, the probability of taking action given the current state of the agent is small. Ishii et al. (2002) assess their proposed method in a small 2D maze environment with fixed and moving barriers and a zig-zag maze. In both

experiments, the effect of exploration bonus on the required number of steps for reaching the shortest path is shown.

Lopes et al. (2012) improve the traditional model-based exploration techniques based on OFU principle such as R-Max (Brafman and Tennenholtz, 2002) and Bayesian exploration bonus (Kolter and Ng, 2009), and empirically estimate the learner's accuracy and the learning progress. Lopes et al. (2012) use the change in the loss of prediction error (both mean and variance) as the bonus term, and subsequently use it to modify R-Max and Bayesian exploration bonus methods. This approach is particularly useful in scenarios where the agent has an incorrect prior knowledge of the transition model, or when it changes over time. The learning progress is measured with respect to the empirical estimate of predictive error using the leave-one-out cross-validation estimator.

6.3.2 PREDICTION ERROR-BASED BONUS: FUNCTION APPROXIMATION

In this section, we focus on the exploration techniques that use PE as a measure of uncertainty to design exploration bonuses in domains with large or infinite state spaces, where methods that focus on tabular settings fail to generalize.

Stadie et al. (2015) propose a method of exploration that hinges upon a model of system dynamics trained using past experiences and observations. A state is considered novel and accordingly receives an exploration bonus based on its disagreement with the environment's learned model. Formally, given the state encoding function σ, the prediction error with respect to σ at time t is denoted by $e_t(\sigma)$ and thus the bonus term is proportional to $e_t(\sigma)/t$. The authors benchmark their proposed algorithm on Atari tasks and show that it is an efficient method of assigning exploration bonuses for large and complex RL domains. The predictive model introduced by Stadie et al. (2015) is a simple two-layer neural network, and the prediction error is measured with respect to the l_2 Euclidean norm. They evaluate their proposed method in 14 Arcade Learning Environments (ALE) against Boltzmann, DQN and Thompson sampling methods.

In another diversity-driven method, Hong et al. (2018) augment a diversity-based bonus with the loss function and encourage the agent to explore diverse behaviours during the training phase. The modified loss function is computed by subtracting the current policy's expected deviation or distance from a set of recently adopted policies. They use a clipping threshold in the case of observing extraordinary deviation in the computed empirical expectation. The authors evaluate the performanc of their proposed method in Mujoco and Atari environments against vanilla DDPG and the Parameter-Noise exploration method (Plappert et al., 2018).

Burda et al. (2018a) conducted a large set of experiments on curiosity-driven learning algorithms that work with intrinsic reward mechanisms across 54 different environments. Interestingly, the results presented show the impact of feature learning on better generalizability while using prediction error as a deriving force for exploration. Through the conducted experiments, they also demonstrate the limitations of prediction-based bonus mechanisms.

Some studies measure the observed state's novelty based on the amount of PE the observed state induces on the network that is trained using the agent's past experience. For example, Burda et al. (2018b) introduce a simple notion of exploration bonus, which

is based on the PE induced by the features of observed states and the prediction of the randomly initialized network when the environment is stochastic. Burda et al. (2018b) count four factors as the primary sources of error in prediction, 1) the amount of training data, 2) stochasticity of environment, 3) model misspecification, and 4) learning dynamics. The uncertainty factor considered in their study is based upon the uncertainty quantification method proposed initially by Osband et al. (2018). Burda et al. (2018b) assess their proposed exploration method in the difficult Atari game Montezuma's Revenge and outperforms the state-of-the-art baselines.

A line of research uses information gain (IG) as an exploration bonus (For a more detailed explanation regarding information gain, refer to section 4.2). For instance, Houthooft et al. (2016) propose a curiosity-driven strategy, which uses information gain as a driving force to encourage exploration of actions that lead to states that cause a larger change in the agent's internal model of environment dynamics. The state and action spaces are considered to be continuous. The paper proposes a variational approach to approximate the true posterior, and therefore, the information gain is measured using the KL-divergence between the agent's internal belief over environment dynamics at different time steps. The main challenge in their proposed model is the computation of variational lower-bound. The way Houthooft et al. (2016) compute variational lower-bound, requires the calculation of the posterior probability, which is generally considered to be computationally intractable. The computed variational lower-bound is used to measure the agent's curiosity. Houthooft et al. (2016) assess their proposed algorithm in continuous Mujoco domains, and compare its performance with TRPO, ERWR and REINFORCE.

Another study that uses IG as an exploration bonus is introduced by Kim et al. (2019). The authors apply the information bottleneck (IB) principle (Tishby et al., 2000) to design an exploration bonus to handle the exploration-exploitation trade-off, particularly in distractive environments. The bonus term in Curiosity-Bottleneck (CB) objective (inspired by IB principle) appears in the form of mutual information, measured by KL-divergence between the latent representation of the environment and the input observation. To inspect the performance of the proposed CB method, Kim et al. (2019) perform experiments on three environments: 1) Novelty detection on MNIST and Fashion-MNIST, 2) Treasure Hunt in a grid-world environment, and 3) Atari's Gravitar, Montezuma's Revenge, and Solaris games. The CB performance is compared with the work of Burda et al. (2018b).

7. Deliberate Exploration

The optimal solution to the exploration-exploitation problem is a strategy that yields the highest expected total reward over the entire duration of an agent's interaction with the environment. The problem of finding such an optimal strategy is a meta-problem in itself, where a notion of optimality can be defined with respect to a distribution over the models of the environments the agent is likely to encounter. Which exploration strategy works best depends on the range of environments considered plausible by the agent. If a probability distribution over the environment is known, we can define the optimal exploration strategy as the strategy that yields the highest expected total reward in expectation over this distribution.

If unknown transition and reward model parameters are considered as the unobservable states of the system, then the entire problem can be defined as a specific kind of POMDP where the hidden part is a set of environment parameters and the observable part is the state of the sampled MDP. However, this formulation can have too many belief states and can thus in general not be solved exactly within practicable time. Various sub-fields have focused on different avenues of solving this problem approximately. The Bayesian approach focuses on tackling the full problem, termed Bayes-adaptive MDP, by introducing different approximations to the problem; for example by making prior assumptions about the form of the uncertainty over the unknown model parameters either via function approximation or representing the distribution over environment using a set of samples. On the other hand, meta-learning approaches assume the agent does not directly have access to a distribution over environments but can be trained on samples from the relevant distribution. Various methods, e.g. from model-free reinforcement learning, can then be used to find policies that can effectively embed interaction histories and map them to actions to be taken. These methods tend to aim at finding (locally) optimal solutions to the Bayes-adaptive MDP within a parametrized family of policies.

7.1 Solving the exploration-exploitation trade-off optimally

In the Bayesian approach for RL, a posterior is maintained over the possible models of the environment given the observations so far. As the agent collects more observations, the belief is updated to reflect this new information. Consequently, learning bayes-optimally in an MDP is equivalent to solving for an optimal action selection strategy in a meta-level Markov decision process defined by these belief states. In the following sections, we will focus on a particular formulation of this meta-level problem: the Bayes-Adaptive MDP (Duff, 2002) formulation, where the belief state is given by a current base-level state as well as a posterior distribution over the base-level transition and reward models.

When the dynamics are unknown, the Bayesian RL formulation assumes that the transition model $\mathcal{P}$ is a latent variable according to some prior distribution $P(\mathcal{P})$. Let H_t denote the history of observations up to time t, then the dynamics model is updated according to the Bayes rule $P(\mathcal{P}|H_t) \propto P(H_t|\mathcal{P})P(\mathcal{P})$, in response to the observed transitions. The uncertainty of the model dynamics is handled by transforming it into uncertainty into the augmented state and history space. The actual state of the agent, $\mathcal{S}$, together with the belief state that that consists of parameters defining uncertainty distributions over the transition model, $\mathcal{X}$, comprise the *hyperstate*, $\mathcal{S}^+ = \mathcal{S} \times \mathcal{X}$. The new dynamics model in this new augmented space is defined by $\mathcal{P}^+(s', x', a, s, x)$, that denotes the transition model for hyperstates, conditioned on action a being taken in hyperstate $\langle s, x \rangle$. The reward function for the aurgmented MDP is given by $\mathcal{R}^+(s, x, a) = \mathcal{R}(s, a)$. The new MDP is defined by the tuple of $M^+ = \langle \mathcal{S}^+, \mathcal{A}, \mathcal{P}^+, \mathcal{R}^+, \gamma \rangle$ and is known as the **Bayes-Adaptive MDP (BAMDP)** (Duff, 2002).

The hyperstate is a sufficient statistics for the process evolving under uncertainty, and the Bellman equations formalism used for MDPs also holds true for the generalized hyperstate MDP. The solution to the Bellman equations gives the value function of the hyperstate, and an optimal value function implicitly defines the optimal learning policy, which maps hyperstates to actions. The value function given by Bellman equation for the BAMDP is
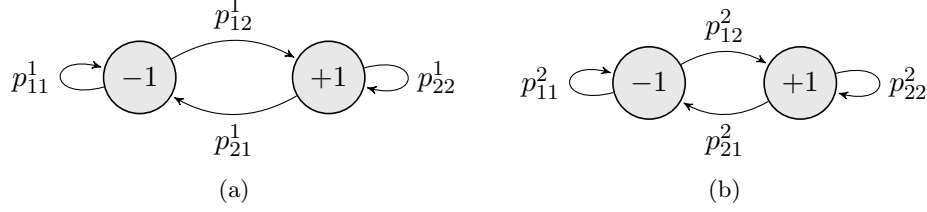
Figure 2: A 2-state MDP with uncertain transition probabilities under (a) action 1 and (b) action 2. Rewards are denoted by ± 1 in the states.

given by:

$$V^\star(s,x) = \max_{a \in \mathcal{A}} \left[\mathcal{R}^+(x,a) + \gamma \sum_{\substack{x' \in \mathcal{X}, \\ s' \in \mathcal{S}}} \mathcal{P}^+(s',x' \mid s,x,a) V^\star(s',x') \right]. \tag{27}$$

The agent starts in the belief state corresponding to its prior and, by executing the greedy policy in the BAMDP while updating its posterior, acts optimally (with respect to its beliefs) in the original MDP. The Bayes-optimal policy for the unknown environment is the optimal policy of the BAMDP, thereby providing an elegant solution to the exploration-exploitation trade-off. Through this framework, rich prior knowledge about the environment can be naturally incorporated into the planning process, potentially leading to more efficient exploration and exploitation of the uncertain world.

Example: We use the following example from Duff (2003) to highlight how the Bayesian formulation allows to solve the exploration-exploitation optimally. Consider an MDP with 2 states and 2 actions as shown in Figure 2. The hyperstate in this case is defined by a tuple $(s;x)$, where the physical state s is the state the agent currently is in as given by the MDP, and the information state x, which is the collection of distributions describing uncertainty in the transition probabilities. The rewards (± 1) are deterministic and are received when the agent lands in the corresponding state. The unknown transition probabilities are denoted by the labelled arcs. For this particular example, the authors propose using an appropriate conjugate family of distributions, such as Dirichlet, to model the uncertainty. For instance, if the uncertainty in p_{11}^1 (transition from state 1 under action 1) is represented as beta distribution parameterized by (α_1^1, β_1^1), then hyper state for $s = 1$ can be written as: $(1; \begin{bmatrix} \alpha_1^1 & \beta_1^1 \\ \beta_2^1 & \alpha_2^1 \end{bmatrix}, \begin{bmatrix} \alpha_1^2 & \beta_1^2 \\ \beta_2^2 & \alpha_2^2 \end{bmatrix})$. The new augmented transition function can be written as

$$P(s',x'|s,x,a) = P(s'|s,x,a)P(x'|s,x,a,s').$$

The first factor can be estimated from samples by taking expectation over the possible transition function to corresponding to a given hyperstate. Therefore, for a transition from s under action a, the corresponding terms for $s' = 1$ and $s' = 2$ will be $\frac{\alpha^a}{\alpha_s^a + \beta_s^a}$ and $\frac{\beta^a}{\alpha_s^a + \beta_s^a}$ respectively. For the second factor, the parameter of the dirichlet is given by the number of 'effective' transitions of each type observed in transit from the initial hyperstate to the given

hyperstate. For this particular example, the form of the posterior update for the information state parameters, given an observation is also a beta distribution, but with parameters that are incremented to reflect the observed data. As such, for transition from s under action a, the update information state terms for $s' = 1$ and $s' = 2$ will be $\alpha_s^a + 1$ and $\beta_s^a + 1$. An optimality equation can be written for the local transitions and the corresponding successor hyperstates. For the above example, the optimal value function has the form:

$$
V\left(1; \begin{bmatrix} \alpha_1^1 & \beta_1^1 \\ \beta_2^1 & \alpha_2^1 \end{bmatrix}, \begin{bmatrix} \alpha_1^2 & \beta_1^2 \\ \beta_2^2 & \alpha_2^2 \end{bmatrix}\right) = \max \left\{ \frac{\alpha_1^1}{\alpha_1^1 + \beta_1^1} \left[r_{11}^1 + V\left(1; \begin{bmatrix} \alpha_1^1 + 1 & \beta_1^1 \\ \beta_2^1 & \alpha_2^1 \end{bmatrix}, \begin{bmatrix} \alpha_1^2 & \beta_1^2 \\ \beta_2^2 & \alpha_2^2 \end{bmatrix}\right) \right] \right.
$$
$$
+ \frac{\beta_1^1}{\alpha_1^1 + \beta_1^1} \left[r_{12}^1 + V\left(2; \begin{bmatrix} \alpha_1^1 & \beta_1^1 + 1 \\ \beta_2^1 & \alpha_2^1 \end{bmatrix}, \begin{bmatrix} \alpha_1^2 & \beta_1^2 \\ \beta_2^2 & \alpha_2^2 \end{bmatrix}\right) \right],
$$
$$
\frac{\alpha_1^2}{\alpha_1^2 + \beta_1^2} \left[r_{11}^2 + V\left(1; \begin{bmatrix} \alpha_1^1 & \beta_1^1 \\ \beta_2^1 & \alpha_2^1 \end{bmatrix}, \begin{bmatrix} \alpha_1^2 + 1 & \beta_1^2 \\ \beta_2^2 & \alpha_2^2 \end{bmatrix}\right) \right]
$$
$$
\left. + \frac{\beta_1^2}{\alpha_1^2 + \beta_1^2} \left[r_{12}^2 + V\left(2; \begin{bmatrix} \alpha_1^1 & \beta_1^1 \\ \beta_2^1 & \alpha_2^1 \end{bmatrix}, \begin{bmatrix} \alpha_1^2 & \beta_1^2 + 1 \\ \beta_2^2 & \alpha_2^2 \end{bmatrix}\right) \right] \right\}.
$$

An optimal policy can be computed using the dynamic programming on the augmented MDP. However, as each transition can increment any of the information state parameters with every time step, there is an exponential increase in number of distinct reachable hyperstates with the time horizon (4^{depth} hyperstates at a given depth for the above example).

Hence, in the Bayesian RL formulation the exploration-exploitation trade-off is handled in a principled manner, because the agent does not heuristically choose between exploiting or exploring, rather, it takes an optimal action (that might lead to a mixture of exploration and exploitation) with respect to its full Bayesian model of the uncertain sequential decision process (Duff, 2002; Poupart et al., 2006). In Bayesian decision theory, the optimal action is the one that, over the entire time horizon considered, maximizes the expected reward, averaged over the possible world models. Any gain in reducing the uncertainty over the posterior transition models is not valued just for its own sake, but rather is driven by the potential gain in the future reward that it offers.

The major problem with the BAMDPs is that the number of hyperstates grows exponentially with the time-horizon. This exponential growth in hyperstates limits the scalability of this approach as it can make solving the problem intractable when either the size of state and action spaces increases or the planning horizon increases. In the next section we will go over some of the works that provide tractable computational procedures that retain the Bayesian formulation but sidesteps the intractability by employing various approximation and sampling techniques.

7.2 Bayesian Methods

To recap, in the Bayes-Adaptive setting, a prior over MDPs is given to the agent, and (approximately) optimal decisions are made based on the posterior over MDPs, where the posterior is a function of the interaction history with the MDP. In this section we review a few notable works that are based on the different BAMDP formulations, to illustrate how the explosion of the hyperstate space is handled to provide tractable approximate solutions.

For a thorough insight into different techniques for tractability in Bayesian methods for RL we refer the reader to the survey by Ghavamzadeh et al. (2015, Chapter 4).

Bayesian Q-Learning (Dearden et al., 1998) adopts a Bayesian approach to the Q-learning by maintaining and propagating the probability distributions to represent the uncertainty over the agent's estimates of Q-values. Under certain modeling assumptions, the authors show that Dirichlet distributions can be used to maintain such a posterior over the Q-values. Instead of solving the BAMDP using dynamic programming as in the previous Section 7.1, the authors propose an action selection procedure based on the 'value of information' - the expected gain in future decision quality that might arise from information acquired from the current action choice. Intuitively, this notion considers the gain that can be acquired learning the true value of a particular Q-value. Formally, let a_1 and a_2 denote the actions with best and second best expected values respectively, and $q_{s,a}^\star$ denote a random variable representing the a possible value of $Q^\star(s, a)$ in some MDP, then the gain from learning the true value is defined as:

$$
Gain_{s,a}(q_{s,a}^\star) = \begin{cases} \mathbb{E}[q(s, a_2)] - q_{s,a}^\star & \text{if } a = a_1 \text{ and } q_{s,a}^\star < \mathbb{E}[q_{s,a_2}], \\ q_{s,a}^\star - \mathbb{E}[q(s, a_1)] & \text{if } a \neq a_1 \text{ and } q_{s,a}^\star > \mathbb{E}[q_{s,a_1}], \\ 0 & \text{otherwise} \end{cases}
\tag{28}
$$

As the agent doesn't know the true value of $q_{s,a}^\star$, the expected gain is computed using the prior beliefs to estimate the Value of Perfect Information (VPI):

$$
VPI(s, a) = \int_{-\infty}^{+\infty} Gain_{s,a}(x) Pr(q_{s,a} = x) dx,
\tag{29}
$$

The value of perfect information gives an upper bound on the myopic (1-step) expected value of information for exploring with action a. In order to take into account the exploitation aspect, the expected reward is also added to the action selection criteria. Therefore, the goal is to select action that maximizes $(\mathbb{E}[q_{s,a}] + VPI(s, a))$.

Once the action is taken and transitions are observed, the authors propose two ways of estimating the distribution of the Q-value. The first one is based on Moment matching that leads to a closed-form update but can become overly confident. The second technique is based on mixture updates that are more cautious but require numerical integration. Finally, they provide some theoretical results on the convergence of the algorithm and then they conclude with some experimental results on three toy problems (a 5-state chain MDP, an 8-state loop MDP and a 2D-maze of size 8x8) and compare their work with three other methods.

Optimal Probe (Duff, 2003) retains the full Bayesian framework but proposes to sidestep the intractable calculations by using a novel actor-critic architecture and proposing a corresponding policy-gradient based update for it. The policies and value functions are approximated by functions involving linear combinations of the information state components. The main assumption is that the value function is a relatively smooth function of the information state, x. For the feature set, they propose using the components of the information state. As such, the critic is parameterized as: $V(s, x) \approx V_s(x) = \sum_l \theta_l[s]x_l$, where V_s represents the function approximator associated with the state in the original MDP s, and θ_l represents the parameters corresponding to the l-th information state component. For policies,

Approach	Choice of approximation	Solution method
Dearden et al. (1998)	Online myopic value function	Dynamic programming
Duff (2002)	Offline value function	Policy gradient
Wang et al. (2005)	Online tree search	Tree backups, myopic heuristic at leaves
Poupart et al. (2006)	Offline value function	Point-based POMDP methods
Guez et al. (2012)	Online tree search	Q-learning with rollout policy

Table 7: Overview of the main techniques covered in Section 7.2

they propose using a separate parameterized function approximator for each original state s and possible action. This is because the stochastic policies that map from hyperstates to actions are required to be relatively smooth over the information state x, but should allow arbitrary and discontinuous variation with respect to the original state and the prospective action. This reduces the size of the class of stochastic policies that the actor can model, but the hope is that this parameterized family of policies will be rich enough to represent *near-optimal* policies.

For updating the policy, a Monte-Carlo based Policy gradient update rule is proposed that uses a single hyperstate trajectory for providing an unbiased estimate of the gradient components with respect to the actor's parameters. In conclusion, the assumed function class for the actor introduces a bias but makes the complexity independent of the (exponential) number of hyperstates. Policy gradients can be expressed in terms of a matrix representing the steady-state probability of hyperstates. This matrix is again computationally intractable but can be approximated using sampled roll-outs. They test their approach a 2-states and 2 actions toy MDP with a horizon of 25 and were the first to show a tractable algorithm for that case. This method however is only computationally feasible for domains with a small number of information states where the proposed architecture works.

Sparse sampling (Kearns et al., 2002) is a sample-based tree search algorithm, where the agent samples the next possible tree nodes from each state and then applies Bellman backup to propagate the values of child nodes to the parent node. In *Bayesian Sparse Sampling* (Wang et al., 2005) the authors apply the sparse sampling technique to search over BAMDPs, where the task is to use lookahead search to estimate the long term value of possible actions in a given belief state. The key idea is to exploit information in the Bayesian posterior to make intelligent action selection decisions during the look-ahead simulation, rather than simply enumerating over all the actions or selecting the actions myopically. The search tree is expanded adaptively in a non-uniform manner, instead of building a uniformly balanced look-ahead tree. The intuition is that the agent only needs to investigate actions that are potentially optimal, and in this way can save computation resources on sub-optimal actions. At each decision node, a promising action is selected using a heuristic based on Thompson sampling (Thompson, 1933) to preferentially expand the tree below actions that appear to be locally promising. At each branch node, a successor belief-state is sampled from the transition dynamics of the belief-state MDP. Once chosen, the action is executed, and a new belief state is entered. As the focus of the work is to demonstrate action selection improvements, they compare their approach with other selection schemes like standard Sparse sampling, Thompson sampling, Interval Estimation, and Boltzmann

exploration techniques in the continuous 2-dimensional action space Gaussian processes task. The results show that their approach yields improved action selection quality whenever Bayesian posteriors can be conveniently calculated.

Poupart et al. (2006), focus on the problem of discrete Bayesian model-based RL in the online setting, and propose the *BEETLE* (Bayesian Exploration Exploitation Tradeoff in LEarning), a point-based value iteration algorithm, that takes into account exploration during the exploitation step itself using the belief states mechanism. The main contribution of the work is that they present an analytical derivation of the optimal value functions for the discrete Bayesian RL problem where the optimal value function is parameterized by a set of multivariate polynomials. This analytical form then allows them to build an efficient point-based value iteration algorithm that exploits the particular form of parameterization. As a result, they have a computationally efficient offline policy optimization technique and results that show the optimization remains efficient as long as the number of unknown transition dynamics parameters remains small. The results are based on the argument that the transition dynamics of many problems can be encoded with few parameters by either tying the parameters together or using a factored model. The algorithm achieves online efficiency, by moving the policy optimization offline and doing only action selection and belief monitoring at run time. The authors compare their proposed method with two-heuristics based exploration approaches on two discrete toy MDPs benchmarks: a toy-chain with 5 states and 2 actions and an assistive technology scenario MDP with 9 stats and 6 actions.

The *Bayes-Adaptive Monte Carlo Planning* (BAMCP) algorithm Guez et al. (2012) provides a sample-based method for approximate Bayes-optimal planning for discrete MDPs that exploits Monte-Carlo tree search (MCTS). The core idea is to use the UCT algorithm (Kocsis and Szepesvári, 2006) in a computationally efficient manner for BAMDPs, where the belief state is approximated using the samples sampled only at the root node of the tree. The authors propose Bayes-Adaptive UCT, where instead of integration over all the transition models, or even approximating this expectation using an average of sampled transition model, only a single transition model $\mathcal{P}^i$ is sampled from the agent's current belief (posterior at the root of the search tree) and is used to simulate all the necessary samples during this episode. A tree policy then treats the forward search as a meta-exploration problem, in a similar manner as the vanilla UCT problem. The goal of the tree policy is to exploit regions of the search tree that appear better than the others while continuing to explore the less known parts of the tree. For the exploitation part, a rollout policy is learned in a model-free manner, using Q-learning, from the samples collected by the agent as a result of the interaction with the environment. In order for further computational efficiency, the authors propose a novel lazy sampling scheme for the partial transition models. The intuition is that if the transition parameters for different states and actions are independent, then instead of sampling a complete $\mathcal{P}$, only the parameters necessary for individual state-action pairs can be sampled. The returns from each episode are then used to update the value of each node in the search tree during the planning. By integrating over many simulations, and therefore many sampled MDPs, the optimal value of each future sequence is obtained with respect to the agent's belief.

The authors compare their method with BOSS (Asmuth et al., 2009) and BEB (Kolter and Ng, 2009) in the discrete grid-world domain (10 x 10 states) and loop domain with 9

states and show their method outperforms the others. They also test their method on an infinite 2D grid-world domain where they show that their method greatly outperforms the other baselines. In the infinite 2D grid domain the baselines can not handle the large state space but as BAMCP limits the posterior inference to the root of the search tree it is not directly affected by the size of the state space, but instead is limited by the planning time.

7.3 Meta-learning

Some exploration strategies are based on, or emerge from, a meta learning perspective. Meta learning focuses on learning an appropriate bias from a collection of tasks that allows more efficient learning on new, similar tasks (Vilalta and Drissi, 2002). The field of meta-reinforcement learning uses this approach in reinforcement learning settings. As such, the agent interacts with multiple train MDPs, allowing it to learn a strategy for interacting with eventual novel test MDPs from the same family. At 'meta-train time' the agent learns general patterns from a set of train tasks, that can be exploited to learn more efficiently at 'meta-test time'. Usually, the train and test tasks are assumed to be drawn from the same distribution. Thus, the agent can tailor the learning strategy employed at 'meta-test time' using inductive biases extracted at 'meta-train time'. As a simple example, the training MDPs might be used to optimize the learning rate used on testing MDPs. However, much more complex meta-strategies might also be learned, including the update rule itself.

Before we give a more detailed breakdown of meta-reinforcement learning methods and exploration strategies used with or emerging from those methods, it is good to realize the connection between meta-reinforcement learning and Bayes-adaptive reinforcement learning. In Bayes-adaptive learning, a prior over MDPs is known to the agent, and (approximately) optimal decisions are made based on the posterior over MDPs. The posterior is a function of the interaction history with the MDP. Compare this set-up with a common set-up for meta-reinforcement learning where train and test tasks are assumed to be drawn from the same distribution. An optimized mapping from the interaction history to the next action to be taken can thus be seen to target the same objective as Bayes-adaptive learning, with the prior represented by a finite set of sampled train MDPs. Thus, meta-reinforcement learning strategies have the potential to learn an approximately optimal exploration-exploitation trade-off with respect to the distribution of training MDPs. Whereas the Bayes-adaptive literature has typically focused on discrete MDPs and tabular representations, most work on meta reinforcement learning focuses on the function approximation case, using deep neural networks to represent policies or value functions.

Meta-reinforcement learning techniques can be classified based on the amount of structure used in defining the mapping from interaction history to actions (Finn, 2018). At the extreme, such policies can be black boxes directly mapping from interaction histories to actions. We can think of this black box as combining two functions usually implemented by separate functions: updating the policy, and executing the policy in the current state. Thus, black box meta-reinforcement learning approaches are sometimes described as learning a reinforcement learning algorithm (Wang et al., 2016a; Duan et al., 2016b).

Other classes of meta-reinforcement learning tasks impose more structure on the mapping from interaction histories to actions. One common type of structure is that the policy update is given by gradient ascent. The agent's objective then becomes finding prior policy

Black box methods	Meta-learned object	
Heess et al. (2015)	Recurrent policy	
Duan et al. (2016b)	Recurrent policy	
Wang et al. (2016a)	Recurrent policy	
Garcia and Thomas (2019)	Adviser policy	
Alet et al. (2020)	Bonus mechanism	
Gradient-based methods	Exploration characteristics	
Finn et al. (2017)	Early gradient-based approach	
Stadie et al. (2018)	More credit to pre-update policies	
Rothfuss et al. (2019)	Low-variance, improved action-level credit assignment.	
Frans et al. (2018)	Efficient exploration through meta-learning sub-policies	
Inference-based methods	Inference type	Use of posterior
Gupta et al. (2018)	Test-time approximate inference	Posterior sampling
Rakelly et al. (2019)	Amortized inference	Posterior sampling
Zintgraf et al. (2019b)	Amortized inference	Conditioning on variational parameters

Table 8: Overview of the main techniques covered in this section.

parameters such that a few gradient steps result in good 'post-update' policies (Finn et al., 2017). Another common type of structure is when, similar to Bayes-adaptive approaches, the inference of the posterior distribution over MDPs is decoupled from the action selection mechanism (Rakelly et al., 2019; Zintgraf et al., 2019a).

Per category, we will now discuss how exploration can be done. In our review, we will focus on papers that explicitly introduce exploration methods or explicitly discuss or analyze exploration behavior in these methods. Table 8 provides an overview of the main methods discussed and some of their characteristics.

Black-box methods In black-box models, a mapping from interaction history to actions is optimized directly. The interaction history could comprise states or observations, actions, and rewards at all previous time steps, or even previous episodes in the same MDP. Since interaction histories are sequences without a fixed size, recurrent neural networks are a popular architecture to represent policies or value functions. Conceptually, these methods are quite straightforward: in theory any well-known policy search method could be used, using a recurrent architecture and with interaction histories rather than single states as input. A generic policy for a black-box method is of the functional form

$$\pi_{\boldsymbol{\theta}}(A_t|S_t, H_t), \quad H_t = [S_0, A_0, R_0, \ldots, S_{t-1}, A_{t-1}, R_{t-1}, S_t],$$

with $\pi_{\boldsymbol{\theta}}$ the (usually recurrent) policy architecture parametrized by $\boldsymbol{\theta}$ and H_t the interaction history up to time t.

Early work on this approach was performed by Heess et al. (2015), who looked at different kinds of partial observability, including a BAMDP-like setting where the agent had to explore a tank of water for a hidden platform, and thereafter exploit this knowledge to find the platform again. Work by Duan et al. (2016b) and Wang et al. (2016a) further investigated this type of approach. These works also formalized the idea of "learning a re-

inforcement learning algorithm". Although the framework is roughly similar, the methods differ in design choices such as which base reinforcement learning method is used (A2C/A3C (Mnih et al., 2016) versus TRPO (Schulman et al., 2015)). Both papers look at the exploration/exploitation trade-off in bandits and visual navigation tasks compared to classical exploration methods based on Thompson sampling or exploration bonuses. On the discrete tasks with known dynamics, the meta-learning approach performs competitively with classical methods, but it is applicable to the challenging visual exploration task where these tabular methods are not applicable. In addition, Duan et al. (2016b) investigate exploration behavior on tabular MDPs, and Wang et al. (2016a) investigate multiple tasks inspired by behavioral science and neuroscience paradigms. Their 'dependent arms' bandit experiments reveal that the method successfully learns the optimal exploration/exploitation strategy for this particular family of bandits, where after one exploration action without any reward the agent switches to pure exploitation behavior. This exploration behavior is not matched by the considered classical bandit algorithms. Overall, the black-box model is very general as well as conceptually straightforward, however, training the recurrent models tends to take a lot of samples and training time.

An interesting approach is that by Garcia and Thomas (2019). Here, an 'advisor' policy is learned in a black-box fashion. Since the black box environment contains a base reinforcement learning algorithm (such as REINFORCE (Williams, 1992) or PPO Schulman et al. (2017)), the meta state becomes a combination of the current MDP, the MDP state, and the base learner's internal state. This structure is reminiscent of gradient-based methods (covered in the next paragraph), although unlike in those methods, gradients are not taken through the internal update of the base learner. Actions executed in the environment are mostly based on those of the base learner, but in a fraction ϵ of time steps an explorative action from the 'advisor' (meta-policy) is executed (rather than an action chosen uniformly as in ϵ-greedy strategies). The authors provide the theoretical result that solving the meta MDP indeed results in optimal exploration policies in the sense of maximizing total return over a set number of episodes averaged over the prior MDP distribution. Furthermore, they show strong empirical performance in the function approximation setting compared to the base algorithms without advisor, random exploration, and the MAML meta-learning approach (Finn et al., 2017, covered in the next section) on continuous control tasks such as the 'Ant' task from the Roboschool environment[5].

Where the methods covered above all meta-learned a policy, Alet et al. (2020) proposed a different approach. Their method meta-learns curiosity mechanisms or bonuses that generalize across very different reinforcement-learning domains. They formulate the problem of finding exploration bonuses as an outer loop that will search over a space of bonuses (meta-learned), and an inner loop that will perform standard reinforcement learning using the adapted reward signal. They propose to do the meta-learning of the bonus in the combinatorial space of programs instead of transferring neural network weights resulting in an approach similar to neural architecture search. The programs are represented in a domain-specific language that includes modular building blocks like neural networks that can be updated with gradient-descent mechanisms, ensembles, buffers, etc. They show that searching through a rich space of programs yields novel designs that work better than human-

---

5. `https://openai.com/blog/roboschool/`

designed methods such as those proposed by Pathak et al. (2017); Burda et al. (2018b). At the same time, the proposed approach generalizes across environments with different state and action spaces, for instance, image-based 2D gridworld games and Mujoco environments like Acrobot.

Gradient-based methods   Gradient-based methods aim to introduce more structure in the meta reinforcement learning problems compared to the discussed black-box methods. These black-box methods in essence learn a RL algorithm specific to the current distribution over MDPs. Thus, it should be no wonder that updates take many steps at meta-train time. Gradient-based methods introduce prior knowledge about typical reinforcement learning updates in the learning process. These methods try to learn a policy in such a way that a few updates (or even a single one) results in a 'post-update' policy that is able to attain high expected returns. As a function of the interaction history, the policy is then of the form

$$\pi_{\boldsymbol{\theta}'(H_t)}(A_t|S_t), \quad \boldsymbol{\theta}'(H_t) \leftarrow \boldsymbol{\theta} + \alpha\nabla_{\boldsymbol{\theta}}\mathbb{E}\left[\sum_{u=0}^{t-1} R(S_u, A_u)\right],$$

with $\boldsymbol{\theta}'$ the post-update parameters, that result from an inner update using an estimate of the policy gradient, and $\mathbf{s}_u, \mathbf{a}_u$ with $u < t$ taken from the interaction history $\mathbf{h}_t$ (Finn et al., 2017).

Stadie et al. (2018) extended earlier work on gradient-based meta-RL (Finn et al., 2017) with the explicit aim to improve exploration. Where the earlier implementation of model-agnostic meta learning (MAML) by Finn et al. (2017) did not properly assign credit to pre-update trajectories, the proposed algorithm was hypothesized to explore better and thus called E-MAML. The proposed approach was tested on benchmark problems including mazes and 'Krazy World', an environment specifically designed to test exploration in meta learning. The proposed approach indeed learned faster on these benchmarks. Furthermore, a separate analysis looked at the difference in exploration metrics (such as the number of goal states visited) and confirmed the proposed algorithm scored higher. Similar results held for E-RL^2, an approach based on RL^2 (Duan et al., 2016b), inspired by E-MAML, which attempts to promote exploration behavior in black-box methods by ignoring rewards obtained during exploratory roll-outs.

MAML and E-MAML were analyzed in more detail by Rothfuss et al. (2019). They found that the MAML formulation takes the internal structure of the policy update better into account. When all terms of the gradient of this formulation are taken into account, one should thus expect better performance. To do so, they propose a low-variance estimator of the required Hessian. Their experiments on various locomotion benchmarks confirm this performance improvement. Furthermore, they explicitly analyze exploration behavior. This qualitative analysis shows that the original MAML implementation does not learn a good exploration strategy, with E-MAML doing better but having a hard time assigning credit to individual actions. The proposed LVC estimator, on the other hand, developed good exploration behavior and was able to exploit the gleaned information.

The approach proposed by Frans et al. (2018) meta-learns a shared hierarchy, meaning that a common set of sub-policies is learned while a master policy that selects between the sub-policies is adapted to the meta-test task at hand. This approach can be compared to methods such as the previously discussed MAML (Finn et al., 2017), although here the

parameters of the shared hierarchy are not updated in the 'inner' update, and second order gradients are not passed back to the 'outer' meta-level updates of the shared hierarchy. Frans et al. (2018) find that the proposed method can successfully learn a meta-structure where exploration takes place efficiently on the level of the master policy. Furthermore, they find that sub-policies learned on small mazes can be transferred effectively to a more challenging sparse-reward navigation task.

Inference-based methods In inference-based methods, the inference of the properties of the current MDP is decoupled from taking actions in that MDP. This structure mirrors classical strategies for solving POMDPs (Monahan, 1982) or BAMDPs (Strens, 2000; Duff, 2002). Since meta-learning methods are often applied to problems with continuous states and non-linear dynamics, inference over tasks can generally not be performed in closed form. Instead, inference-based meta-reinforcement learning algorithms tend to use a learned latent embedding space and often employ some form of approximate inference (Gupta et al., 2018; Rakelly et al., 2019; Zintgraf et al., 2019a). A generic inference-based policy architecture could thus be described as

$$\pi_{\boldsymbol{\theta}}(A_t|S_t, \boldsymbol{\phi}^*), \quad \boldsymbol{\phi}^* = \arg\min_{\boldsymbol{\phi}} \mathrm{KL}(q_{\boldsymbol{\phi}}(Z)||p(Z|H_t)),$$

with ϕ^* the optimal parameters for a variational distribution q over latent context variables Z.

Model agnostic exploration with structured noise (MAESN), finds an approximate posterior using gradient ascent at meta-test time (Gupta et al., 2018). Latent context variables can then be drawn in a posterior-sampling like manner. These latent variables are constant during an episode and thus allow structured exploration behavior. The experience with meta-train tasks is thus used both to initialize a policy and to acquire a latent exploration space that can inject structured stochasticity into a policy. The method is compared experimentally to MAML, RL^2, and conventional (non-meta) learning strategies. The authors find that MAESN strongly outperforms baseline strategies in terms of learning speed and performance after 100 learning iterations. Furthermore, qualitative analysis shows MAESN is able to learn a well-structured latent space that effectively explores in the space of coherent strategies for the trained family of environments.

Both other methods perform inference in this latent space by training an amortized inference network. Rakelly et al. (2019) optimize this network to directly minimize a chosen loss function while staying close to a prior. The resulting posterior is then used for posterior sampling. The authors find improved results compared to earlier meta-learning methods, presumably thanks to the structured and efficient exploration as well as the ability to use off-policy data offered by the decoupling of inference and acting.

The approach by Zintgraf et al. (2019a), instead, explicitly optimizes the embedding space to decode transition and reward information. Also, instead of posterior sampling, the policy is conditioned on the mean and co-variance of the full variational posterior. In preliminary experiments, the authors show that the approximate posterior can be used to strategically and systematically explore gridworlds. In these experiments, the proposed method outperformed black-box meta-learning methods by a large margin.

8. Probability Matching

An entire body of algorithms for efficient exploration is inspired by the Probability Matching approach, also known as Thompson Sampling (Thompson, 1933). Probability matching (or Thompson Sampling) is a heuristic for balancing the exploration-exploitation dilemma in the Multi-Arm Bandit setting (Li and Chapelle, 2012; Agrawal and Goyal, 2012). In this setting, the agent maintains a posterior distribution over its beliefs regarding the optimal action, but instead of selecting the action with the highest expected return according to the belief posterior, the agent selects the action randomly according to the probability with which it deems that action to be optimal. This approach uses the variance of the posterior to induce randomization and incentivizes the exploration of uncertain states and actions. As more experience is gathered, the variance of the posterior will decrease and concentrate on the true value. Thompson sampling is provably efficient for the bandit setting (Russo and Van Roy, 2013).

We use the setting from Agrawal and Goyal (2012) to give an example of the Thompson Sampling algorithm for Bernoulli bandit setting, i.e. when the agent gets a binary reward (0 or 1) for selecting an arm i, and the probability of success is μ_i. The algorithm maintains Bayesian priors on the Bernoulli means μ_i. The algorithm initially assumes arm i to have a uniform prior on μ_i (`Beta(1, 1)`). At time t, having observed $S_i(t)$ successes and $F_i(t)$ failures plays of the arm i, the algorithm corresponding updates distribution on μ_i as `Beta`$(S_i(t) + 1, F_i(t) + 1)$. The algorithm then samples the model of the means from these posterior distributions of the μ_i's and plays an arm according to the probability of its mean being the largest.

Posterior Sampling for Reinforcement Learning (PSRL) *Bayesian dynamic programming* was first introduced in Strens (2000) and is more recently known as *posterior sampling for reinforcement learning* (PSRL) (Osband et al., 2013). PSRL can be thought of as an extension of the Thompson Sampling algorithm to the RL setting with finite state and action spaces. Compared to Thompson Sampling, where a model is re-sampled at every time-step[6], PSRL samples a single model for an episode and follows this policy for the duration of the episode.

In PSRL, the agent starts with a prior belief over the model of the MDP and then proceeds to update its full posterior distribution over models with the newly observed samples. For each episode, a model hypothesis is then sampled from this distribution, and traditional planning techniques are used to solve the MDP and obtain the optimal value function. For the current episode, the agent follows the greedy policy with respect to the optimal value function. They evaluate their approach on a 6-state chain MDP with 3 actions and a random MDP with 10 state and 5 actions. They show that PSRL outperforms UCRL2 by a large margin in both the above domains.

Although, both categories of the methods maintain a distribution over the rewards and transition dynamics obtained using a Bayesian modeling approach, PSRL based methods employ the posterior sampling exploration algorithm that requires solving for an optimal policy for a single MDP in each iteration. As such, PSRL is more computationally efficient compared to typical Bayesian-Adaptive algorithms that find optimal exploration strategy

---

6. In the bandit setting the length of an episode is 1 time-step.

via either dynamic programming or tree look-ahead in the Bayesian belief state space over a set of a prior distribution over MDPs.

Best of Sampled Set (BOSS) (Asmuth et al., 2009) drives exploration by sampling multiple models from the posterior and combining them to select actions optimistically. The proposed algorithm resembles RMAX (Brafman and Tennenholtz, 2002) in the sense that it samples multiple models from the posterior only when the number of transitions from a state-action pair exceeds a certain threshold. The sampled models are then merged into an optimistic MDP which is solved to select the best action. They show that this approach leads to sufficient exploration to guarantee finite-sample performance guarantees. They compare their approach against BEETLE and RMAX and show superior results on the 5-state chain problem and 6x6 grid-world.

Agrawal and Jia (2017) propose an algorithm based on posterior sampling that achieves near-optimal worst-case regret bounds when the underlying MDP is communicating with (unknown) finite diameter. The diameter D is defined as an upper bound on the time it takes to move from any state s to any other s' using an appropriate policy, for each pair of s, s'. The algorithm combines the optimism in the face of uncertainty principle (Section 6) with the posterior sampling heuristic. The algorithm proceeds in epochs, where, at the beginning of each epoch the algorithm generates $\psi = \tilde{O}(S)$ sample transition probability vectors from a posterior distribution for every state and action. It then proceeds to solve the extended MDP with ψA actions and S states formed using these samples. The optimal policy found from the extended MDP is then used for the entire epoch. This algorithm can be viewed as a combination of methodologies from BOSS and PSRL algorithms described above. The main contribution of this work is providing tighter regret bounds, and as such, they do not provide any experimental results for their algorithm.

Randomized Value Functions The PSRL approach is limited to the finite state and action setting, where learning the model and planning might be tractable. For the rest of the section, we will look at the approaches that aren't based on modeling the MDP transition and rewards explicitly, but instead focus on estimating distributions over the value functions directly. The underlying assumption of these approaches is that approximating the posterior distribution for the value function is more statistically and computationally efficient than learning the MDP. Osband et al. (2016c, 2017) proposed a family of methods called *Randomized Value Functions* (RVFs) in order to improve the scalability of PSRL. At an abstract level, RVFs can be interpreted as a model-free version of PSRL. These methods directly model a distribution over the value functions instead of over MDPs. The agent then works by sampling a randomized value function at the beginning of each episode and following that for the rest of the episode. Exploring with dithering strategies (Sec. 4.1, 5, 5.2), is inefficient as the agent may oscillate back and forth, it might not be able to discover temporally extended interesting behaviours. On the other hand, exploring with Randomized Value Functions, the agent is committed to a randomized but internally consistent strategy for the entire length of the episode. The switch to value function modelling also facilitates the use of function approximation.

In order to scale posterior sampling approach to large MDPs with *linear* function approximation, Osband et al. (2016b) introduce *Randomized Least Square Value Iteration (RLSVI)* that involves using Bayesian linear regression for learning the value function. The goal is to

extend PSRL to value function learning, that would involve maintaining a belief distribution over candidates for the optimal value function. Before each episode, the agent would then sample a value function from its posterior distribution and then apply the associated greedy policy throughout the episode.

Least Square Value Iteration (LSVI) (Sutton and Barto, 1998a; Szepesvári, 2010) performs a linear regression for the Bellman error at each timestep - similar to Fitted Q-Iteration (Riedmiller, 2005). As the value function learned from LSVI has no notion of uncertainty, algorithms based on just LSVI have to rely on other exploration strategies, like blind exploration (Sec. 4.1). RLSVI also performs linear regression for one-step Bellman error but it incorporates a Gaussian uncertainty estimate for the resultant value function. This is equivalent to replacing the linear regression step of LSVI with a Bayesian linear regression as if the one-step Bellman error was sampled from a Gaussian distribution. Even though this is not the correct Bayesian distribution, Osband et al. (2016b) show that it is still useful for approximating the uncertainty. As the RLSVI updates the value function based on a random sample from this distribution, the resultant value function is also a random sample from the approximate posterior. RLSVI is a provably efficient algorithm for exploration in large MDPs with linear value function approximators (Osband et al., 2017). The authors compare their approach with the dithering based strategies in a didactic chain environment where RLVSI is able to scale up to 100 state length chain. They also show better learning performance of RLVSI compared LSPI and LSVI on learning to play Tetris task, and a recommendation engine task, both of which have exponential state space but they have access to the appropriate basis functions for the task.

One of the problems with this approach is that the distributions over the value functions can be as complex to represent as distributions over transition model, and exact Bayesian inference might not be computationally tractable. RLSVI does not explicitly maintain and update belief distributions, as a coherent Bayesian method would, but still serves as a computationally tractable method for sampling value functions.

Osband et al. (2016a) propose bootstrapped Q-learning, an RVF based approach, as an extension of RLSVI to nonlinear function approximators. Bootstrapped-DQN consists of a simple non-parametric bootstrap[7] with random initialization to generate approximate posterior samples over Q-values. This technique helps in the scenario where exact Bayesian inference is intractable, such as in deep networks. Bootstrapped-DQN consists of a network with K bootstrapped estimates of the Q-function, trained in parallel. This means that each $Q_1, \ldots, Q_K$ provide a temporally extended (and consistent) estimate of the value uncertainty. At the start of each episode, the agent samples one head, which it follows for the duration of the episode. Bootstrapped-DQN is a non-parametric approach to uncertainty estimation. The authors also show that, when used with deep neural networks, the bootstrap can produce reasonable estimates of uncertainty. They compare their method against DQN Mnih et al. (2015) on the didactic chain MDP with 100 states, and on the Atari domain on the Arcade Learning Environment (Bellemare et al., 2013), where they show that Bootstrapped-DQN is able to learn faster and also improves the final score in most of the games.

Many other exploration methods, such as (Azizzadenesheli et al., 2018; Touati et al., 2019) can be interpreted as combining the concept of RVF with neural network function

---

7. Bootstrap uses the empirical distribution of a sampled dataset as an estimate of the population statistic.

Approach	Posterior Sampling	Theoretical properties
Strens (2000); Osband et al. (2013); Osband and Van Roy (2017)	Sample 1 MDP model per episode	Bounded Expected regret $\tilde{O}(HS\sqrt{AT})$, Bounded Bayesian regret $\tilde{O}(H\sqrt{SAT})$
Asmuth et al. (2009)	Sample K MDP models per step	PAC-MDP
Osband et al. (2016b)	Sample 1 value function per episode	Bounded Expected regret $\tilde{O}(\sqrt{H^3SAT})$
Agrawal and Jia (2017)	Sample $\tilde{O}(S)$ transition probability vectors per epoch	Bounded Worst-case regret $\tilde{O}(D\sqrt{SAT})$
Osband et al. (2016a)	Sample 1 head Q-network from the bootstrap ensemble per episode	-
Azizzadenesheli et al. (2018)	Sample weights for last layer of Q-network per episode	-
Touati et al. (2019)	Sample noise variables for the normalizing flow per episode	-
Janz et al. (2019)	Sample weights for the Q-network based on the successor features for every episode	-

Table 9: Overview of the main techniques covered in Section 8. For the theoretical properties column, S and A denote the cardinalities of the state and action spaces, T denotes time elapsed, H denotes the episode duration, and D denotes the diameter.

approximation. It allows these methods to scale to high-dimensional problems, such as Atari domain (Bellemare et al., 2013), that otherwise might be too computationally extensive for PSRL. However, the approximations introduced in these works come with trade-offs that are not present in the original PSRL work. Specifically, because a value function is defined with respect to a particular policy, constructing posterior over the value functions requires selection of a reference policy or distribution over policies. However, in practice, the above methods do not enforce any explicit structure or dependencies. Janz et al. (2019) propose Successor Uncertainties (SU), a scalable and computationally cheaper (compared to Bootstrapped DQN) model-free exploration algorithm that retains the key elements of the PSRL. SU models the posterior over rewards and state transitions directly and derives the posterior over the value functions analytically, thereby ensuring the posterior over the values estimates matches the posterior sampling policy. Empirically, SU performs much better on hard tabular didactic chain problem where the algorithm scales up to chains of length 200 states. On the Atari domain, SU outperforms the closest RVF algorithm - Bootstrapped-DQN on 36 of 49 games.

9. Conclusion and Perspectives

In this survey, we have proposed a categorization for reinforcement learning exploration strategies based on the information that the agent uses in its exploratory action selection. We divided exploration techniques into two general classes: "Reward-Free Exploration" and "Reward-Based Exploration". The *reward-free* techniques either select actions totally at random, or utilize some notion of intrinsic information in order to guide exploration, without taking into account extrinsic rewards. This can be useful in environments where the reward signal is very sparse, and therefore not immediately available to the agent. *Reward-based* exploration methods leverage the information related to the reward signal and can further be divided into the "memory-free", which only take into account the state of the environment, and "memory-based", which consider additional information about the history of the agent's interaction with the environment. In each of these categories, methods which share similar properties are clustered together in one subcategory. The basis for this clustering is mainly the type of information used, as well as the way it is used in the selection of exploratory actions. We have discussed these exploration methods and have pointed out their strengths and limitations, as well as improvements that have been made and some which are still possible. We would like to emphasize that our goal was not to review the theoretical sample complexity results, which are abundant in the field. Rather, we wanted to provide a big picture which captures the current "lay of the land in terms of methods", and which is useful for practitioners in their choice of methods. We note that theoretical results often need to rely on assumptions about the environment and the RL algorithm, for example tabular or linear representations of the value function, or smooth dynamics, which are often not satisfied in practice. Nonetheless, exploration methods can still provide useful empirical results even if their theoretical assumptions are not satisfied.

In this study, we have limited ourselves to sequential decision making in MDPs. We have not covered in detail strategies for POMDPs and bandits, although these settings have provided inspiration for some of methods proposed for the MDP setting. An emerging trend is the study of safe exploration methods, but as relatively little is written on them so far, we

have not focused on this topic. Finally, considering the large number of yearly publications in this field, we have excluded some methods that are similar to the major classes of approaches we discussed.

Owing to the improved and more accessible computational power in the recent years, the newly proposed exploration techniques have contributed significantly to the improvement in the exploration-exploitation dilemma. However, there are major concerns and issues that have not been resolved yet, mainly due to the absence of a consensus over the ways exploration methods can be assessed. For instance, different techniques are evaluated according to different measures of efficiency and performance, such as state coverage, information gain, sample efficiency, or regret. Furthermore, there is no set of standard experimental tasks that all proposed exploration techniques are evaluated on. This diversity in the methods of assessment complicates the comparison of exploration techniques together. Finally, there is often some sort of a discrepancy between the theoretical guarantees that a method provides and the experimental condition the agent encounters. Consequently, there is often no reliable guarantee for the performance of these techniques in more involved environments. Addressing these issues could be the focus of future work.

10. Acknowledgement

References

Youssef Achbany, Francois Fouss, Luh Yen, Alain Pirotte, and Marco Saerens. Optimal tuning of continual online exploration in reinforcement learning. In *International Conference on Artificial Neural Networks*, pages 790–800. Springer, 2006.

Shipra Agrawal and Navin Goyal. Analysis of thompson sampling for the multi-armed bandit problem. In *Conference on Learning Theory*, pages 39–1, 2012.

Shipra Agrawal and Randy Jia. Optimistic posterior sampling for reinforcement learning: worst-case regret bounds. In *Advances in Neural Information Processing Systems*, pages 1184–1194, 2017.

David J Aldous. Exchangeability and related topics. In *École d'Été de Probabilités de Saint-Flour XIII—1983*, pages 1–198. Springer, 1985.

Ferran Alet, Martin F Schneider, Tomas Lozano-Perez, and Leslie Pack Kaelbling. Meta-learning curiosity algorithms. In *International Conference on Learning Representations*, 2020.

Teresa M Amabile, William DeJong, and Mark R Lepper. Effects of externally imposed deadlines on subsequent intrinsic motivation. *Journal of personality and social psychology*, 34(1):92, 1976.

Susan Amin, Maziar Gomrokchi, Hossein Aboutalebi, Harsh Satija, and Doina Precup. Locally persistent exploration in continuous control tasks with sparse rewards. *arXiv preprint arXiv:2012.13658*, 2020.

Charles W Anderson. *Learning and problem solving with multilayer connectionist systems*. PhD thesis, University of Massachusetts at Amherst, 1986.

András Antos, Csaba Szepesvári, and Rémi Munos. Fitted q-iteration in continuous action-space mdps. In *Advances in neural information processing systems*, pages 9–16, 2008.

John Asmuth, Lihong Li, Michael L Littman, Ali Nouri, and David Wingate. A Bayesian sampling approach to exploration in reinforcement learning. In *Conference on Uncertainty in Artificial Intelligence*, pages 19–26, 2009.

Peter Auer and Ronald Ortner. Logarithmic online regret bounds for undiscounted reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 49–56, 2007.

Mohammad Gheshlaghi Azar, Vicenç Gómez, and Bert Kappen. Dynamic policy programming with function approximation. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 119–127, 2011.

Mohammad Gheshlaghi Azar, Ian Osband, and Rémi Munos. Minimax regret bounds for reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 263–272. JMLR. org, 2017.

Kamyar Azizzadenesheli, Emma Brunskill, and Animashree Anandkumar. Efficient exploration through bayesian deep q-networks. In *Information Theory and Applications Workshop (ITA)*, pages 1–9. IEEE, 2018.

J Andrew Bagnell and Jeff Schneider. Covariant policy search. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2003.

Peter L Bartlett and Ambuj Tewari. Regal: A regularization based algorithm for reinforcement learning in weakly communicating mdps. *arXiv preprint arXiv:1205.2661*, 2012.

Andrew G Barto, Richard S Sutton, and Charles W Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE transactions on systems, man, and cybernetics*, 13(5):834–846, 1983.

Andrew Gehret Barto, Steven J Bradtke, and Satinder P Singh. *Real-time learning and control using asynchronous dynamic programming*. University of Massachusetts at Amherst, Department of Computer and Information Science, 1991.

Joan Bas-Serrano, Sebastian Curi, Andreas Krause, and Gergely Neu. Logistic *q*-learning. Technical Report arXiv:2010.11151, arXiv, 2020.

Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems*, pages 1471–1479, 2016.

Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

Carl A Benware and Edward L Deci. Quality of learning with an active versus passive motivational set. *American Educational Research Journal*, 21(4):755–765, 1984.

Ronen I Brafman and Moshe Tennenholtz. R-max-a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3(Oct): 213–231, 2002.

John S Bridle. Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters. In *Advances in neural information processing systems*, pages 211–217, 1990.

Sébastien Bubeck, Rémi Munos, and Gilles Stoltz. Pure exploration in multi-armed bandits problems. In *International conference on Algorithmic learning theory*, pages 23–37. Springer, 2009.

Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A Efros. Large-scale study of curiosity-driven learning. In *International Conference on Learning Representations*, 2018a.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018b.

Scott Burlington and Gregory Dudek. Spiral search as an efficient mobile robotic search technique. In *Proceedings of the 16th National Conf. on AI, Orlando Fl*, 1999.

Olivier Caelen and Gianluca Bontempi. Improving the exploration strategy in bandit algorithms. In *International Conference on Learning and Intelligent Optimization*, pages 56–68. Springer, 2007.

Pierguido VC Caironi and Marco Dorigo. Training q-agents, 1994.

Moses S Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 380–388, 2002.

Kamil Ciosek, Quan Vuong, Robert Loftin, and Katja Hofmann. Better exploration with optimistic actor critic. In *Advances in Neural Information Processing Systems*, pages 1785–1796, 2019.

Cédric Colas, Olivier Sigaud, and Pierre-Yves Oudeyer. GEP-PG: Decoupling exploration and exploitation in deep reinforcement learning algorithms. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1039–1048, 2018.

Edoardo Conti, Vashisht Madhavan, Felipe Petroski Such, Joel Lehman, Kenneth O Stanley, and Jeff Clune. Improving exploration in evolution strategies for deep reinforcement learning via a population of novelty-seeking agents. In *Advances in Neural Information Processing Systems*, 2018.

Will Dabney, Georg Ostrovski, and André Barreto. Temporally-extended {\epsilon}-greedy exploration. *arXiv preprint arXiv:2006.01782*, 2020.

Peter Dayan. Improving generalization for temporal difference learning: The successor representation. *Neural Computation*, 5(4):613–624, 1993.

Peter Dayan and Terrence J Sejnowski. Exploration bonuses and dual control. *Machine Learning*, 25(1):5–22, 1996.

Richard Dearden, Nir Friedman, and Stuart Russell. Bayesian q-learning. In *AAAI National Conference on Artificial Intelligence*, pages 761–768, 1998.

Richard Dearden, Nir Friedman, and David Andre. Model based Bayesian exploration. In *Conference on Uncertainty in Artificial Intelligence*, pages 150–159. Morgan Kaufmann Publishers Inc., 1999.

Edward Deci and Richard M Ryan. *Intrinsic motivation and self-determination in human behavior*. Springer Science & Business Media, 1985.

Edward L Deci. Effects of externally mediated rewards on intrinsic motivation. *Journal of personality and Social Psychology*, 18(1):105, 1971.

Edward L Deci. *Intrinsic motivation*. Plenum Press., New York, NY, US, 1975. ISBN 978-1-4613-4448-3.

Marc Peter Deisenroth, Gerhard Neumann, Jan Peters, et al. A survey on policy search for robotics. *Foundations and Trends® in Robotics*, 2(1–2):1–142, 2013.

Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International Conference on Machine Learning*, pages 1329–1338, 2016a.

Yan Duan, John Schulman, Xi Chen, Peter L Bartlett, Ilya Sutskever, and Pieter Abbeel. RL^2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016b.

Michael O Duff. Design for an optimal probe. In *International Conference on Machine Learning*, pages 131–138, 2003.

Michael O'Gordon Duff. *Optimal Learning: Computational procedures for Bayes-adaptive Markov decision processes*. PhD thesis, University of Massachusetts at Amherst, 2002.

Eyal Even-Dar and Yishay Mansour. Convergence of optimistic and incremental q-learning. In *Advances in neural information processing systems*, pages 1499–1506, 2002.

Sarah Filippi, Olivier Cappé, and Aurélien Garivier. Optimism in reinforcement learning and kullback-leibler divergence. In *2010 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 115–122. IEEE, 2010.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135, 2017.

Chelsea B Finn. *Learning to Learn with Gradients*. PhD thesis, University of California, Berkeley, 2018.

Sébastien Forestier, Yoan Mollard, and Pierre-Yves Oudeyer. Intrinsically motivated goal exploration processes with automatic curriculum learning. *arXiv preprint arXiv:1708.02190*, 2017.

Meire Fortunato, Mohammad Gheshlaghi Azar, Bilal Piot, Jacob Menick, Matteo Hessel, Ian Osband, Alex Graves, Volodymyr Mnih, Remi Munos, Demis Hassabis, Olivier Pietquin, Charles Blundell, and Shane Legg. Noisy networks for exploration. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rywHCPkAW.

Kevin Frans, Jonathan Ho, Xi Chen, Pieter Abbeel, and John Schulman. Meta learning shared hierarchies. In *International Conference on Learning Representations*, 2018.

Ronan Fruit, Matteo Pirotta, Alessandro Lazaric, and Ronald Ortner. Efficient bias-span-constrained exploration-exploitation in reinforcement learning. In *Proceedings of the International Conference on Machine Learning*, volume 80, pages 1573–1581, 2018.

Justin Fu, John Co-Reyes, and Sergey Levine. Ex2: Exploration with exemplar models for deep reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 2577–2587, 2017. URL https://arxiv.org/abs/1703.01260.

Zoltán Gábor, Zsolt Kalmár, and Csaba Szepesvári. Multi-criteria reinforcement learning. In *ICML*, volume 98, pages 197–205, 1998.

Francisco M Garcia and Philip S Thomas. A meta-mdp approach to exploration for lifelong reinforcement learning. In *Advances in neural information processing systems*, 2019.

Matthieu Geist, Bruno Scherrer, and Olivier Pietquin. A theory of regularized markov decision processes. Technical Report arXiv:1901.11275, arXiv, 2019.

Mohammad Ghavamzadeh, Shie Mannor, Joelle Pineau, Aviv Tamar, et al. Bayesian reinforcement learning: A survey. *Foundations and Trends® in Machine Learning*, 8(5-6): 359–483, 2015.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

Geoffrey J Gordon. Stable function approximation in dynamic programming. In *Machine Learning Proceedings 1995*, pages 261–268. Elsevier, 1995.

Robert Grande, Thomas Walsh, and Jonathan How. Sample efficient reinforcement learning with gaussian processes. In *International Conference on Machine Learning*, pages 1332–1340, 2014.

Wendy S Grolnick and Richard M Ryan. Autonomy in children's learning: An experimental and individual difference investigation. *Journal of personality and social psychology*, 52 (5):890, 1987.

Arthur Guez, David Silver, and Peter Dayan. Efficient bayes-adaptive reinforcement learning using sample-based search. In *Advances in Neural Information Processing Systems*, pages 1025–1033, 2012.

Vijaykumar Gullapalli. A stochastic reinforcement learning algorithm for learning real-valued functions. *Neural networks*, 3(6):671–692, 1990.

Ying Guo, Astrid Zeman, and Rongxin Li. A reinforcement learning approach to setting multi-objective goals for energy demand management. *International Journal of Agent Technologies and Systems (IJATS)*, 1(2):55–70, 2009.

Zhaohan Guo and Emma Brunskill. Concurrent PAC RL. In *AAAI*, pages 2624–2630, 2015.

Abhishek Gupta, Russell Mendonca, Yuxuan Liu, Pieter Abbeel, and Sergey Levine. Meta-reinforcement learning of structured exploration strategies. In *Advances in Neural Information Processing Systems*, pages 5302–5311, 2018.

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *Proceedings of the International Conference on Machine Learning*, 2017.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pages 1861–1870, 2018.

Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary computation*, 9(2):159–195, 2001.

Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 2094–2100, 2016.

Elad Hazan, Sham M Kakade, Karan Singh, and Abby Van Soest. Provably efficient maximum entropy exploration. In *International Conference on Machine Learning*, pages 2681–2691, 2019.

Nicolas Heess, Jonathan J Hunt, Timothy P Lillicrap, and David Silver. Memory-based control with recurrent neural networks. *arXiv preprint arXiv:1512.04455*, 2015.

Zhang-Wei Hong, Tzu-Yun Shann, Shih-Yang Su, Yi-Hsiang Chang, Tsu-Jui Fu, and Chun-Yi Lee. Diversity-driven exploration strategy for deep reinforcement learning. *Advances in Neural Information Processing Systems*, 31:10489–10500, 2018.

Rein Houthooft, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. Vime: Variational information maximizing exploration. In *Advances in Neural Information Processing Systems*, pages 1109–1117, 2016.

Shin Ishii, Wako Yoshida, and Junichiro Yoshimoto. Control of exploitation–exploration meta-parameter in reinforcement learning. *Neural networks*, 15(4-6):665–687, 2002.

Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(Apr):1563–1600, 2010.

David Janz, Jiri Hron, Przemysław Mazur, Katja Hofmann, José Miguel Hernández-Lobato, and Sebastian Tschiatschek. Successor uncertainties: exploration and uncertainty in temporal difference learning. In *Advances in Neural Information Processing Systems*, pages 4509–4518, 2019.

Chi Jin, Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael I Jordan. Is q-learning provably efficient? In *Advances in Neural Information Processing Systems*, pages 4863–4873, 2018.

Yuu Jinnai, Jee Won Park, David Abel, and George Konidaris. Discovering options for exploration by minimizing cover time. *arXiv preprint arXiv:1903.00606*, 2019.

Yuu Jinnai, Jee Won Park, Marlos C Machado, and George Konidaris. Exploration in reinforcement learning with deep covering options. In *International Conference on Learning Representations*, 2020.

Nicholas K Jong and Peter Stone. Model-based exploration in continuous state spaces. In *International Symposium on Abstraction, Reformulation, and Approximation*, pages 258–272. Springer, 2007.

Michael I Jordan. Generic constraints on underspecified target trajectories. In *International Joint Conference on Neural Networks*, volume 1, pages 217–225. IEEE Press New York, 1989.

Leslie Pack Kaelbling. *Learning in embedded systems*. MIT press, 1993.

Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

Sham Kakade, Michael J Kearns, and John Langford. Exploration in metric state spaces. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 306–312, 2003.

Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine learning*, 49(2-3):209–232, 2002.

Michael Kearns, Yishay Mansour, and Andrew Y Ng. A sparse sampling algorithm for near-optimal planning in large markov decision processes. *Machine learning*, 49(2-3):193–208, 2002.

Michał Kempka, Marek Wydmuch, Grzegorz Runc, Jakub Toczek, and Wojciech Jaśkowski. Vizdoom: A doom-based ai research platform for visual reinforcement learning. In *2016 IEEE Conference on Computational Intelligence and Games (CIG)*, pages 1–8. IEEE, 2016.

Youngjin Kim, Wontae Nam, Hyunwoo Kim, Ji-Hoon Kim, and Gunhee Kim. Curiosity-bottleneck: Exploration by distilling task-specific novelty. In *International Conference on Machine Learning*, pages 3379–3388, 2019.

Levente Kocsis and Csaba Szepesvári. Bandit based monte-carlo planning. In *European conference on machine learning*, pages 282–293. Springer, 2006.

Nate Kohl and Peter Stone. Policy gradient reinforcement learning for fast quadrupedal locomotion. In *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, volume 3, pages 2619–2624. IEEE, 2004.

J Zico Kolter and Andrew Y Ng. Near-Bayesian exploration in polynomial time. In *International Conference on Machine Learning*, pages 513–520. ACM, 2009.

Jan Koutnik, Faustino Gomez, and Jürgen Schmidhuber. Evolving neural networks in compressed weight space. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, pages 619–626. ACM, 2010.

Raksha Kumaraswamy, Matthew Schlegel, Adam White, and Martha White. Context-dependent upper-confidence bounds for directed exploration. In *Advances in Neural Information Processing Systems*, pages 4779–4789, 2018.

Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

Kyungjae Lee, Sungjoon Choi, and Songhwai Oh. Sparse markov decision processes with causal sparse tsallis entropy regularization for reinforcement learning. *IEEE Robotics and Automation Letters*, 3(3):1466–1473, 2018.

Douglas B Lenat. AM: An artificial intelligence approach to discovery in mathematics as heuristic search. Technical report, Stanford university, department of computer science, 1976.

Lihong Li and Olivier Chapelle. Open problem: Regret bounds for thompson sampling. In *Conference on Learning Theory*, pages 43–1, 2012.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations*, 2016.

Long-Ji Lin. Self-improving reactive agents: Case studies of reinforcement learning frameworks. In *Proceedings of the International Conference on Simulation of Adaptive Behavior: From Animals to Animats*, 1990.

Long-Ji Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine learning*, 8(3-4):293–321, 1992.

Daniel Ying-Jeh Little and Friedrich Tobias Sommer. Learning and exploration in action-perception loops. *Frontiers in neural circuits*, 7:37, 2013.

Chunming Liu, Xin Xu, and Dewen Hu. Multiobjective reinforcement learning: A comprehensive overview. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(3): 385–398, 2015.

Manuel Lopes, Tobias Lang, Marc Toussaint, and Pierre-Yves Oudeyer. Exploration in model-based reinforcement learning by empirically estimating learning progress. In *Advances in Neural Information Processing Systems*, pages 206–214, 2012.

Marlos C Machado, Marc G Bellemare, and Michael Bowling. A Laplacian framework for option discovery in reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning*, pages 2295–2304, 2017.

Marlos C Machado, Marc G Bellemare, and Michael Bowling. Count-based exploration with the successor representation. Technical Report arXiv:1807.11622, arXiv, 2018a.

Marlos C. Machado, Clemens Rosenbaum, Xiaoxiao Guo, Miao Liu, Gerald Tesauro, and Murray Campbell. Eigenoption discovery through the deep successor representation. In *International Conference on Learning Representations*, 2018b. URL `https://openreview.net/forum?id=Bk8ZcAxR-`.

Marlos C Machado, Marc G Bellemare, and Michael Bowling. Count-based exploration with the successor representation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5125–5133, 2020.

Sridhar Mahadevan. Proto-value functions: Developmental reinforcement learning. In *Proceedings of the 22nd international conference on Machine learning*, pages 553–560. ACM, 2005.

Sridhar Mahadevan and Jonathan Connell. Scaling reinforcement learning to robotics by exploiting the subsumption architecture. In *Machine Learning Proceedings 1991*, pages 328–332. Elsevier, 1991.

Sridhar Mahadevan and Jonathan Connell. Automatic programming of behavior-based robots using reinforcement learning. *Artificial intelligence*, 55(2-3):311–365, 1992.

Anuj Mahajan, Tabish Rashid, Mikayel Samvelyan, and Shimon Whiteson. Maven: Multiagent variational exploration. In *Advances in Neural Information Processing Systems*, pages 7611–7622, 2019.

Jarryd Martin, S Suraj Narayanan, Tom Everitt, and Marcus Hutter. Count-based exploration in feature space for reinforcement learning. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 2471–2478, 2017.

William L Maxwell and Jack A Muckstadt. Design of automatic guided vehicle systems. *Iie Transactions*, 14(2):114–124, 1982.

Nicolas Meuleau and Paul Bourgine. Exploration of multi-state environments: Local measures and back-propagation of uncertainty. *Machine Learning*, 35(2):117–154, 1999.

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

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937, 2016.

Shariq A Mobin, James A Arnemann, and Fritz Sommer. Information-based learning by agents in unbounded state spaces. In *Advances in Neural Information Processing Systems*, pages 3023–3031, 2014.

George E Monahan. State of the art—a survey of partially observable Markov decision processes: theory, models, and algorithms. *Management Science*, 28(1):1–16, 1982.

Andrew W Moore and Christopher G Atkeson. Prioritized sweeping: Reinforcement learning with less data and less time. *Machine learning*, 13(1):103–130, 1993.

Andrew William Moore. *Efficient memory-based learning for robot control*. PhD thesis, University of Cambridge, 1990.

Jun Morimoto and Kenji Doya. Acquisition of stand-up behavior by a real robot using hierarchical reinforcement learning. *Robotics and Autonomous Systems*, 36(1):37–51, 2001.

Michael C Mozer and Jonathan Bachrach. Discovering the structure of a reactive environment by exploration. In *Advances in neural information processing systems*, pages 439–446, 1990.

Paul Munro. A dual back-propagation scheme for scalar reward learning. In *Annual Conference of the Cognitive Science Society*, pages 165–176, 1987.

Ofir Nachum, Haoran Tang, Xingyu Lu, Shixiang Gu, Honglak Lee, and Sergey Levine. Why does hierarchy (sometimes) work so well in reinforcement learning? Technical Report 1909.10618, arXiv, 2019.

Arun Nair, Praveen Srinivasan, Sam Blackwell, Cagdas Alcicek, Rory Fearon, Alessandro De Maria, Vedavyas Panneershelvam, Mustafa Suleyman, Charles Beattie, Stig Petersen, et al. Massively parallel methods for deep reinforcement learning. *arXiv preprint arXiv:1507.04296*, 2015.

Gergely Neu, Anders Jonsson, and Vicenç Gómez. A unified view of entropy-regularized Markov decision processes. Technical Report 1705.07798, arXiv, 2017.

Andrew Y Ng and Michael Jordan. Pegasus: A policy search method for large MDPs and POMDPs. In *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, pages 406–415. Morgan Kaufmann Publishers Inc., 2000.

Derrick Nguyen and Bernard Widrow. The truck backer-upper: An example of self-learning in neural networks. In *Advanced neural computers*, pages 11–19. Elsevier, 1990.

Ali Nouri and Michael L Littman. Multi-resolution exploration in continuous spaces. In *Advances in neural information processing systems*, pages 1209–1216, 2009.

Ronald Ortner and Daniil Ryabko. Online regret bounds for undiscounted continuous reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 1763–1771, 2012.

Ian Osband and Benjamin Van Roy. Why is posterior sampling better than optimism for reinforcement learning? In *International Conference on Machine Learning*, pages 2701–2710, 2017.

Ian Osband, Daniel Russo, and Benjamin Van Roy. (More) efficient reinforcement learning via posterior sampling. In *Advances in Neural Information Processing Systems*, pages 3003–3011, 2013.

Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped DQN. In *Advances in neural information processing systems*, pages 4026–4034, 2016a.

Ian Osband, Benjamin Van Roy, and Zheng Wen. Generalization and exploration via randomized value functions. In *International Conference on Machine Learning*, pages 2377–2386, 2016b.

Ian Osband, Benjamin Van Roy, and Zheng Wen. Generalization and exploration via randomized value functions. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 2377–2386. JMLR.org, 2016c.

Ian Osband, Daniel Russo, Zheng Wen, and Benjamin Van Roy. Deep exploration via randomized value functions. *arXiv preprint arXiv:1703.07608*, 2017.

Ian Osband, John Aslanides, and Albin Cassirer. Randomized prior functions for deep reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 8617–8629, 2018.

Georg Ostrovski, Marc G Bellemare, Aaron van den Oord, and Rémi Munos. Count-based exploration with neural density models. *arXiv preprint arXiv:1703.01310*, 2017.

Pierre-Yves Oudeyer, Frdric Kaplan, and Verena V Hafner. Intrinsic motivation systems for autonomous mental development. *IEEE transactions on evolutionary computation*, 11(2): 265–286, 2007.

Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning (ICML)*, volume 2017, 2017.

Jason Pazis and Ronald Parr. Pac optimal exploration in continuous space Markov decision processes. In *AAAI*, 2013.

Julien Perez, Cécile Germain-Renaud, Balázs Kégl, and Charles Loomis. Responsive elastic computing. In *Proceedings of the 6th international conference industry session on Grids meets autonomic computing*, pages 55–64. ACM, 2009.

Jan Peters, Katharina Mulling, and Yasemin Altun. Relative entropy policy search. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

Matthias Plappert, Rein Houthooft, Prafulla Dhariwal, Szymon Sidor, Richard Y. Chen, Xi Chen, Tamim Asfour, Pieter Abbeel, and Marcin Andrychowicz. Parameter space noise for exploration. In *International Conference on Learning Representations*, 2018.

Pascal Poupart, Nikos Vlassis, Jesse Hoey, and Kevin Regan. An analytic solution to discrete Bayesian reinforcement learning. In *Proceedings of the International Conference on Machine Learning*, pages 697–704. ACM, 2006.

Kate Rakelly, Aurick Zhou, Deirdre Quillen, Chelsea Finn, and Sergey Levine. Efficient off-policy meta-reinforcement learning via probabilistic context variables. *arXiv preprint arXiv:1903.08254*, 2019.

Tabish Rashid, Bei Peng, Wendelin Boehmer, and Shimon Whiteson. Optimistic exploration even with a pessimistic initialisation. In *International Conference on Learning Representations*, 2020.

Alfréd Rényi et al. On measures of entropy and information. In *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*. The Regents of the University of California, 1961.

Martin Riedmiller. Neural fitted q iteration–first experiences with a data efficient neural reinforcement learning method. In *European Conference on Machine Learning*, pages 317–328. Springer, 2005.

Justinian P Rosca. Entropy-driven adaptive representation. In *Proceedings of the workshop on genetic programming: From theory to real-world applications*, volume 9, pages 23–32. Citeseer, 1995.

Jonas Rothfuss, Dennis Lee, Ignasi Clavera, Tamim Asfour, and Pieter Abbeel. Promp: Proximal meta-policy search. In *International Conference on Learning Representations*, 2019.

Thomas Rückstiess, Frank Sehnke, Tom Schaul, Daan Wierstra, Yi Sun, and Jürgen Schmidhuber. Exploring parameter space in reinforcement learning. *Paladyn*, 1(1):14–24, 2010.

Daniel Russo and Benjamin Van Roy. Eluder dimension and the sample complexity of optimistic exploration. In *Advances in Neural Information Processing Systems*, pages 2256–2264, 2013.

Richard M Ryan and Edward L Deci. Intrinsic and extrinsic motivations: Classic definitions and new directions. *Contemporary educational psychology*, 25(1):54–67, 2000.

Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017.

Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim GJ Rudner, Chia-Man Hung, Philip HS Torr, Jakob Foerster, and Shimon Whiteson. The starcraft multi-agent challenge. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pages 2186–2188. International Foundation for Autonomous Agents and Multiagent Systems, 2019.

Jürgen Schmidhuber. Making the world differentiable: On using self-supervised fully recurrent neural networks for dynamic reinforcement learning and planning in non-stationary environments. Technical Report FKI-126-90, Technische Universität München, 1990.

Jürgen Schmidhuber. Curious model-building control systems. In *Neural Networks, 1991. 1991 IEEE International Joint Conference on*, pages 1458–1463. IEEE, 1991a.

Jürgen Schmidhuber. A possibility for implementing curiosity and boredom in model-building neural controllers. In *Proc. of the international conference on simulation of adaptive behavior: From animals to animats*, pages 222–227, 1991b.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. Technical Report 1707.06347, arXiv, 2017.

Paul D Scott and Shaul Markovitch. Learning novel domains through curiosity and conjecture. In *IJCAI*, pages 669–674, 1989.

Frank Sehnke, Christian Osendorfer, Thomas Rückstieß, Alex Graves, Jan Peters, and Jürgen Schmidhuber. Parameter-exploring policy gradients. *Neural Networks*, 23(4):551–559, 2010.

Tim Seyde, Wilko Schwarting, Sertac Karaman, and Daniela L Rus. Learning to plan via deep optimistic value exploration. In Alexandre M. Bayen, Ali Jadbabaie, George Pappas, Pablo A. Parrilo, Benjamin Recht, Claire Tomlin, and Melanie Zeilinger, editors, *Conference on Learning for Dynamics and Control*, volume 120 of *Proceedings of Machine Learning Research*, pages 815–825. PMLR, 10–11 Jun 2020.

Pranav Shyam, Wojciech Jaśkowski, and Faustino Gomez. Model-based active exploration. In *International Conference on Machine Learning*, pages 5779–5788, 2019.

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International Conference on International Conference on Machine Learning*, 2014.

Satinder Pal Singh. Transfer of learning by composing solutions of elemental sequential tasks. *Machine Learning*, 8(3-4):323–339, 1992.

Bradly C. Stadie, Sergey Levine, and Pieter Abbeel. Incentivizing exploration in reinforcement learning with deep predictive models. Technical Report 1507.00814, arXiv, 2015. URL http://arxiv.org/abs/1507.00814.

Bradly C. Stadie, Ge Yang, Rein Houthooft, Xi Chen, Yan Duan, Yuhuai Wu, Pieter Abbeel, and Ilya Sutskever. Some considerations on learning to explore via meta-reinforcement learning. In *ICLR Workshop track*, 2018.

Jan Storck, Sepp Hochreiter, and Jürgen Schmidhuber. Reinforcement driven information acquisition in non-deterministic environments. In *Proceedings of the international conference on artificial neural networks, Paris*, volume 2, pages 159–164. Citeseer, 1995.

Alexander Strehl and Michael Littman. Exploration via model based interval estimation. In *International Conference on Machine Learning*. Citeseer, 2004.

Alexander L Strehl and Michael L Littman. An analysis of model-based interval estimation for Markov decision processes. *Journal of Computer and System Sciences*, 74(8):1309–1331, 2008.

Alexander L Strehl, Lihong Li, Eric Wiewiora, John Langford, and Michael L Littman. Pac model-free reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 881–888, 2006.

Malcolm Strens. A Bayesian framework for reinforcement learning. In *International Conference on Machine Learning*, pages 943–950, 2000.

Freek Stulp and Olivier Sigaud. Path integral policy improvement with covariance matrix adaptation. In *International Conference on Machine Learning*, 2012.

Richard S Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Machine Learning Proceedings 1990*, pages 216–224. Elsevier, 1990.

Richard S Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *ACM SIGART Bulletin*, 2(4):160–163, 1991a.

Richard S Sutton. Integrated modeling and control based on reinforcement learning and dynamic programming. In *Advances in neural information processing systems*, pages 471–478, 1991b.

Richard S Sutton. Reinforcement learning architectures. In *Proceedings ISKIT'92 International Symposium on Neural Information Processing*. Citeseer, 1992.

Richard S Sutton. Generalization in reinforcement learning: Successful examples using sparse coarse coding. In *Advances in neural information processing systems*, pages 1038–1044, 1996.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. Cambridge, MA: MIT Press, 1998a.

Richard S Sutton and Andrew G Barto. Reinforcement learning: an introduction mit press. *Cambridge, MA*, 1998b.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

Csaba Szepesvári. Algorithms for reinforcement learning. *Synthesis lectures on artificial intelligence and machine learning*, 4(1):1–103, 2010.

István Szita and András Lőrincz. The many faces of optimism: a unifying approach. In *Proceedings of the 25th international conference on Machine learning*, pages 1048–1055, 2008.

Prasad Tadepalli and DoKyeong Ok. Model-based average reward reinforcement learning. *Artificial intelligence*, 100(1-2):177–224, 1998.

Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, OpenAI Xi Chen, Yan Duan, John Schulman, Filip DeTurck, and Pieter Abbeel. # exploration: A study of count-based exploration for deep reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 2750–2759, 2017.

Mandayam A.L. Thathachar and P.S. Sastry. A class of rapidly converging algorithms for learning automata. In *IEEE International Conference on Cybernetics and Society*, pages 602–606, 1984.

Evangelos Theodorou, Jonas Buchli, and Stefan Schaal. A generalized path integral control approach to reinforcement learning. *Journal of Machine Learning Research*, 11(Nov): 3137–3181, 2010.

William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 1933.

Sebastian Thrun, Knut Möller, and Alexander Linden. Planning with an adaptive world model. In *Advances in neural information processing systems*, pages 450–456, 1991.

Sebastian B Thrun. Efficient exploration in reinforcement learning. Technical Report CMU-CS-92-102, Carnegie-Mellon University, 1992.

Sebastian B Thrun and Knut Möller. Active exploration in dynamic environments. In *Advances in neural information processing systems*, pages 531–538, 1992.

Arryon D Tijsma, Madalina M Drugan, and Marco A Wiering. Comparing exploration strategies for q-learning in random stochastic mazes. In *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–8. IEEE, 2016.

Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.

Michel Tokic. Adaptive ε-greedy exploration in reinforcement learning based on value differences. In *Annual Conference on Artificial Intelligence*, pages 203–210. Springer, 2010.

Michel Tokic and Günther Palm. Value-difference based exploration: adaptive control between epsilon-greedy and softmax. In *Annual Conference on Artificial Intelligence*, pages 335–346. Springer, 2011.

Aristide Tossou, Debabrota Basu, and Christos Dimitrakakis. Near-optimal optimistic reinforcement learning using empirical bernstein inequalities. *arXiv preprint arXiv:1905.12425*, 2019.

Ahmed Touati, Harsh Satija, Joshua Romoff, Joelle Pineau, and Pascal Vincent. Randomized value functions via multiplicative normalizing flows. *UAI*, 2019.

Long Tran-Thanh, Archie Chapman, Enrique Munoz de Cote, Alex Rogers, and Nicholas R Jennings. Epsilon–first policies for budget–limited multi-armed bandits. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

Peter Vamplew, Richard Dazeley, and Cameron Foale. Softmax exploration strategies for multiobjective reinforcement learning. *Neurocomputing*, 263:74–86, 2017.

Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. In *Advances in neural information processing systems*, pages 4790–4798, 2016.

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

Herke van Hoof, Daniel Tanneberg, and Jan Peters. Generalized exploration in policy search. *Machine Learning*, 106(9-10):1705–1724, 2017.

Kristof Van Moffaert, Madalina M Drugan, and Ann Nowé. Hypervolume-based multi-objective reinforcement learning. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 352–366. Springer, 2013a.

Kristof Van Moffaert, Madalina M Drugan, and Ann Nowé. Scalarized multi-objective reinforcement learning: Novel design techniques. In *2013 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, pages 191–199. IEEE, 2013b.

Joannes Vermorel and Mehryar Mohri. Multi-armed bandit algorithms and empirical evaluation. In *European conference on machine learning*, pages 437–448. Springer, 2005.

Ricardo Vilalta and Youssef Drissi. A perspective view and survey of meta-learning. *Artificial intelligence review*, 18(2):77–95, 2002.

Jane X Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Remi Munos, Charles Blundell, Dharshan Kumaran, and Matt Botvinick. Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763*, 2016a.

Tao Wang, Daniel Lizotte, Michael Bowling, and Dale Schuurmans. Bayesian sparse sampling for on-line reward optimization. In *Proceedings of the 22nd international conference on Machine learning*, pages 956–963. ACM, 2005.

Yuanhao Wang, Kefan Dong, Xiaoyu Chen, and Liwei Wang. Q-learning with ucb exploration is sample efficient for infinite-horizon mdp. In *International Conference on Learning Representations*, 2020.

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, and Nando Freitas. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, pages 1995–2003. PMLR, 2016b.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

Christopher John Cornish Hellaby Watkins. *Learning from delayed rewards*. PhD thesis, King's College, Cambridge, 1989.

Pawel Wawrzynski. Control policy with autocorrelated noise in reinforcement learning for robotics. *International Journal of Machine Learning and Computing*, 5(2):91, 2015.

Martha White and Adam White. Interval estimation for reinforcement-learning algorithms in continuous-state domains. In *Advances in Neural Information Processing Systems*, pages 2433–2441, 2010.

Steven D Whitehead. A complexity analysis of cooperative mechanisms in reinforcement learning. In *AAAI*, pages 607–613, 1991.

Steven D Whitehead and Dana H Ballard. Learning to perceive and act by trial and error. *Machine Learning*, 7(1):45–83, 1991.

Marco Wiering and Jürgen Schmidhuber. Efficient model-based exploration. In *Proceedings of the Sixth International Conference on Simulation of Adaptive Behavior: From Animals to Animats*, volume 6, pages 223–228, 1998.

Marco A Wiering. *Explorations in efficient reinforcement learning*. PhD thesis, University of Amsterdam, 1999.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

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

Yifan Wu, George Tucker, and Ofir Nachum. The laplacian in rl: Learning representations with efficient approximations. *arXiv preprint arXiv:1810.04586*, 2018.

Jeremy Wyatt. *Exploration and inference in learning from reinforcement*. PhD thesis, University of Edinburgh. College of Science and Engineering. School of Informatics., 1998.

Tianbing Xu, Qiang Liu, Liang Zhao, and Jian Peng. Learning to explore with meta-policy gradient. In *Proceedings of the International Conference on Machine Learning*, volume 80, pages 5459–5468, 2018.

Peng-Yeng Yin. Maximum entropy-based optimal threshold selection using deterministic reinforcement learning with controlled randomization. *Signal Processing*, 82(7):993–1006, 2002.

Alexander Zimin and Gergely Neu. Online learning in episodic markovian decision processes by relative entropy policy search. In *Advances in Neural Information Processing Systems 26*, pages 1583–1591, 2013.

Luisa Zintgraf, Maximilian Igl, Kyriacos Shiarlis, Anuj Mahajan, Katja Hofmann, and Shimon Whiteson. Variational task embeddings for fast adaptation in deep reinforcement learning. In *Workshop on "Structure & Priors in Reinforcement Learning" at ICLR*, 2019a.

Luisa Zintgraf, Kyriacos Shiarli, Vitaly Kurin, Katja Hofmann, and Shimon Whiteson. Fast context adaptation via meta-learning. In *International Conference on Machine Learning*, volume 97, pages 7693–7702, Long Beach, California, USA, 2019b. PMLR.