A Review of Reinforcement Learning Evolution: Taxonomy, Challenges and Emerging Solutions

Ji Loun Tan[1], Bakr Ahmed Taha[2], Norazreen Abd Aziz[3], Mohd Hadri Hafiz Mokhtar[4], Muhammad Mukhlisin[5], Norhana Arsad[6]*

Department of Electrical, Electronic and Systems Engineering, Faculty of Engineering and Built Environment, Universiti Kebangsaan Malaysia, Bangi 43600, Selangor, Malaysia[1, 2, 3, 4, 6]

Department of Civil Engineering, Politeknik Negeri Semarang, Jl. Prof. Soedarto SH, Tembalang, Semarang, Jawa Tengah 50275, Indonesia[5]

***Abstract*—Reinforcement Learning (RL) has become a rapidly advancing field inside Artificial Intelligence (AI) and self-sufficient structures, revolutionizing the manner in which machines analyze and make selections. Over the past few years, RL has advanced notably with the improvement of more sophisticated algorithms and methodologies that address increasingly complicated actual-world troubles. This progress has been driven by using enhancements in computational power, the availability of big datasets, and improvements in machine-getting strategies, permitting RL to address challenges across a wide range of industries, from robotics and autonomous driving system to healthcare and finance. The effect of RL is evident in its capacity to optimize selection-making procedures in unsure and dynamic environments. By getting to know from interactions with the environment, RL agents can make decisions that maximize lengthy-time period rewards, adapting to converting situations and enhancing over time. This adaptability has made RL an invaluable tool in situations wherein traditional approaches fall brief, especially in complicated, excessive-dimensional spaces and behind-schedule remarks. This review aims to offer radical information about the current nation of RL, highlighting its interdisciplinary contributions and how it shapes the destiny of AI and autonomous technologies. It discusses how RL affects improvements in robotics, natural language processing, and recreation while exploring its deployment's ethical and practical demanding situations. Additionally, it examines key research from numerous fields that have contributed to RL's development.**

***Keywords*—Artificial intelligence; autonomous systems; decision-making optimization; reinforcement learning; robotics**

I. Introduction

Machine Learning (ML) is primarily categorized into three main types, which are Supervised Learning, Unsupervised Learning, and Reinforcement Learning (RL) [1, 2]. The primary goal of RL is to allow machines to acquire knowledge beyond the constraints of supervised and unsupervised learning paradigms [3]. RL commonly employs a reward function as a training mechanism for agents tasked with specific objectives [4]. Unlike other ML paradigms that rely on labeled datasets, RL derives knowledge through direct interaction with the environment [5]. To make the RL result more effective, it is very important to ensure communication between the agents and the environment [6].

Historically, RL has evolved from early work in behavioral psychology and control theory to become a fundamental tool in artificial intelligence and robotics. The foundational work of Kaelbling et al. (1996) and subsequent advances such as deep Q-networks from Mnih et al. (2015) have set the foundation for developments of RL in future [4, 7]. Hence, nowadays RL research ranges from autonomous robots to complex decision-making systems. For example, RL is able to play an important role in improving robot autonomy and flexibility, especially in tasks like manipulation and navigation [8]. In addition, RL has proven valuable in enhancing autonomous vehicle control, improving safety and optimizing transportation systems [9, 10] The application of RL is not only limited to robotics, but also been applied in the semiconductor industry, where it can optimize processes such as physical design routing [11, 12].

In recent years, the combination of reinforcement learning and deep learning, which is also known as deep reinforcement learning (DRL) [13, 14, 15]. DRL has driven to breakthroughs in solving high-dimensional problems, especially in game-playing AI such as Alpha Go and autonomous systems [16, 17]. In addition, emerging trends nowadays include multi-agent reinforcement learning (MARL) which multiple agents learn and collaborate in a shared environment and also healthcare applications, where RL able to shows promise in personalized medicine and treatment planning [18, 19].

Furthermore, current RL research is more likely to focus on improving sample efficiency, safety, and scalability for real-world applications. It is very important to investigate new ways to integrate Reinforcement Learning with other machine learning paradigms to produce more adaptive and generalizable AI systems. This review will explore the development of RL, classification of RL method, and highlight its modern applications and future research directions. Moreover, this review also will discuss the research contributions from various fields to describe the current state of Reinforcement Learning and its potential to drive innovation in artificial intelligence and autonomous systems. The applications of RL are shown in Fig. 1.

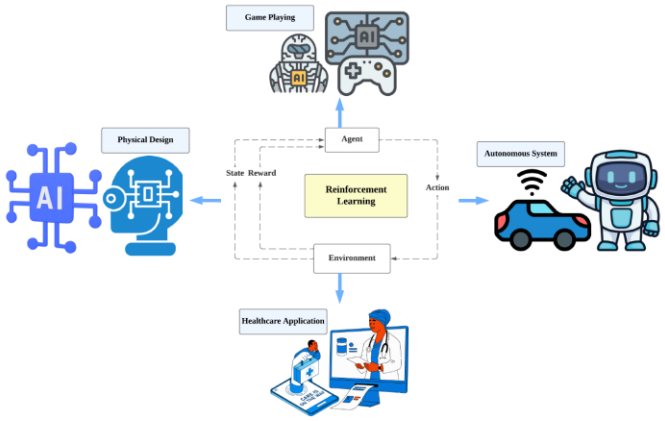

Fig. 1. Applications of reinforcement learning.

In this review, we have a look at the speedy improvement of RL and its developing applicability to complex, actual global troubles. We begin by exploring the evolution of RL techniques, from foundational strategies to advanced strategies consisting of Deep Reinforcement Learning and multi-agent systems. It also categorizes RL techniques, distinguishing between model-unfastened and model-based totally tactics, and highlights their respective strengths and barriers. Furthermore, we cover various RL programs across numerous domain names, including self-sustaining structures, robotics, healthcare, and synthetic intelligence. This exploration aims to offer a comprehensive understanding of the contemporary state of RL, its interdisciplinary contributions, and its potential to pressure destiny innovations in AI and self-reliant technologies.

II. Emerging Trends of Reinforcement Learning Evolution

The evolution of Reinforcement Learning (RL) is based in multiple foundational fields, which include behavioral psychology, trial-and-error learning, optimal control theory and dynamic programming. These parallel developments have provided the foundation for modern RL and shaping its principles and algorithms.

A. Behavioural Psychology and Trial-and-Error Learning

The earliest roots of Reinforcement Learning (RL) can be traced to behavioral psychology, specifically the works of Edward Thorndike and B.F. Skinner before the timeframe of 1960s [20, 21, 22]. Thorndike's Law of Effect introduced the concept of learning from the consequences of actions, where actions followed by satisfying outcomes are more likely to be repeated [20, 23]. This was the basis for trial-and-error learning, which is one of the important aspects of RL, where an agent explores different actions and adapts its behavior based on rewards or punishment. Furthermore, B.F. Skinner demonstrated that operant behavior can be shaped through reinforcement mechanisms, which highlighted the importance of rewards and punishments in learning [24]. This psychological perspective shows the foundation for how RL agents learn to optimize their actions by maximizing rewards or minimizing punishment.

B. Optimal Control Theory

During the mid-20th century, there were major advances in optimal control theory, especially in the field of engineering [25]. Control theory usually focuses on designing controllers that can guide dynamic systems to perform specific tasks in an optimal manner. The Bellman equations which were proposed by Richard Bellman in 1957 became central to this optimal control framework [26], [27], [28]. This work on dynamic programming provided a way to decompose complex decision problems into simpler subproblems, which enables the computation of optimal policies in environments with known dynamics and become a foundation for RL. Hence, Richard Bellman's work directly influenced by introducing the concept of a value function and estimates the expected future reward of being in a particular state and taking a particular action. This concept is basic for the RL algorithms such as Q-learning and value iteration [29].

C. Dynamic Programming and Modern Developments

From the foundation of Bellman's dynamic programming as mentioned, Ronald Howard introduced Markov decision processes (MDP) which is a mathematical framework for modeling decisions in environments where outcomes are partially random and partially controlled by the decision maker [30]. The formalization of MDP set the foundation for modern RL algorithms, as it represents the interaction between an agent and its environment, where the agent seeks to maximize a cumulative reward over time, positive rewards are awarded for favorable actions, while negative rewards or punishments are assigned for undesirable actions. These reward mechanisms are used as evaluative feedback and enable the agent to assess its actions within a specific state and learn from accumulated experiences. However, dynamic programming methods require information or knowledge of the dynamics of the environment, this disadvantage limits its applicability to real world problems. This gap flattens the way for RL techniques. For example, the model-free learning which will be discussed in next section, where an agent can learn optimal policies directly from interactions with the environment without required the knowledge of its dynamics.

Around the 1980s, RL emerged as a distinct field. For example, Sutton introduced temporal difference (TD) learning which is one of the key innovations that enabled agents to learn value functions from incomplete trajectories, rather than waiting until the end of an episode [31]. Furthermore, Watkins further advanced RL by allowing agents to directly learn action-value functions without the requirement for explicit models of the environment [32].

D. Deep Reinforcement Learning (DRL) Revolution

In the 2010s, the combination of Deep Learning (DL) and Reinforcement Learning (RL) which is known as Deep Reinforcement Learning (DRL) revolutionized the field again. In 2015, Google DeepMind researchers introduced the Deep Q Network (DQN) in 2015, which enabled reinforcement learning agents to handle complex tasks such as Atari 2600 games by using deep neural networks to approximate value functions [4]. This research highlighted the scalability of RL in higher dimensional state spaces and lead to major achievements such as the success of AlphaGo, which defeated

the one of the world champions of Go Lee Se-dol by using a combination of RL and DL [33].

E. Emerging Trends and Future Directions

Recent trends in Reinforcement Learning (RL) focus on improving sample efficiency, safety and scalability for real-world applications [34, 35]. New RL techniques such as Multi-agent Reinforcement Learning (MARL), hierarchical reinforcement learning and transfer learning are being explored to solve the issue of complex multi-agent environments where multiple agents learn collaboratively at the same time [18, 36, 37, 38]. In addition, RL can also be applied in more applications in different areas such as robotics, healthcare, finance and autonomous systems. The timeline of key evolution in RL development is shown in Fig. 2.

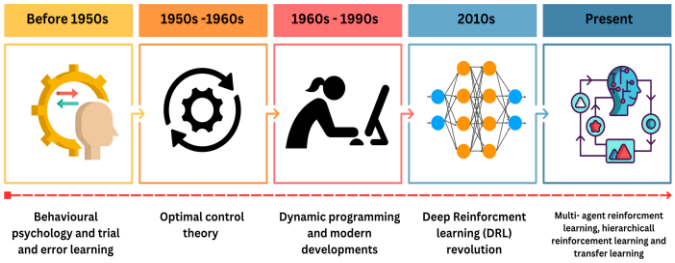

Fig. 2. Timeline of key evolution in RL development.

III. Taxonomy and Criteria of Reinforcement Learning

Reinforcement Learning (RL) techniques are primarily classified into Model-Based and Model-Free approaches within the framework of Markov Decision Processes (MDP). Model-based RL is further categorized into Given-the-Model and Learn-the-Model techniques. Meanwhile, Model-Free RL is subdivided into on-policy and off-policy approaches, which are discussed in the subsequent sections. In addition, value-based and policy-based approaches also have been discussed in this paper. The overview of RL classification is shown in Fig. 3.

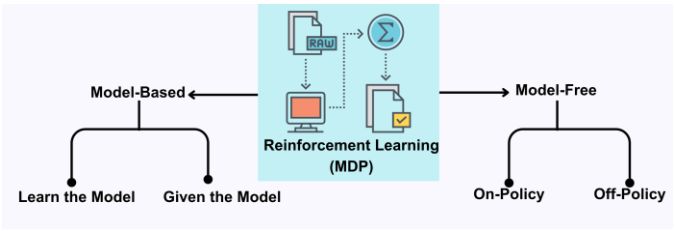

Fig. 3. Overview of reinforcement learning classification.

A. Model-based and Model-Free

Reinforcement Learning (RL) typically demands a substantial amount of data to attain satisfactory performance levels. This section will primarily focus on two types of RL algorithms which include model-free and model-based [39, 40]. Generally, model-free RL algorithms are considered as a direct approach, while model-based RL algorithms are viewed as an indirect method [41].

Model-free RL algorithms aim to learn a policy or value function without explicitly constructing a model of the control system [42]. In contrast, model-based RL algorithms not only learn a value and policy function but also simultaneously construct an explicit model of the system [7]. There are two well-known model-free RL algorithms which are Q-Learning and Deep Q-Networks (DQN), where the agent learns value functions that estimate the expected cumulative rewards for each action in each state [4, 43, 44]. Based on the research conducted by Atkeson and Santamaria, a comparative study was undertaken using a linear double integrator movement task to assess data efficiency, the research findings indicate that the model-based RL algorithms surpass the model-free RL algorithms in terms of data efficiency [45].

In addition, the model-free RL algorithms do not train a model of the environment and aim to directly assign values to states or state-action [46, 47]. The agent directly interacts with the environment and enhances its performance based on the collected samples through exploration. It is easier to implement as they do not require explicit modeling of the environment but might be a problem that is hard to learn. However, model-free RL algorithms are usually hard to implement in real-world scenarios due to the time consumption and the cost [48]. This model-based RL is usually more suitable for large, complex environments but suffers from sample inefficiency as the agent learns only through interactions with the environment.

The advantage of a model-based RL algorithm includes its ability to predict future states and rewards through the explicit modelling of the environment [48, 49]. This will help the agent in making better planning and incorporating strategies like pure planning and expert iterations [50]. In model-based RL, the agent can simulate possible scenarios and plan its actions accordingly. However, model-based RL algorithms come with few disadvantages. One of the major challenges of model-based RL is that the model often depends on the accuracy of the transition model, it means that inaccurate models can lead to domain shift and poor performance [49, 51, 52, 53]. For model-based RL, developing and maintaining an accurate model of the environment can be complex and resource-intensive. Besides, the learned models may be inaccurate in practical scenarios, introducing bias in estimation [51]. When policy estimation and improvement are based on a biased model, the resulting policies may prove ineffective or even collapse when applied in the real environment. Hence, model-based methods often require significant computational cost for model learning and policy optimization hence it is limited application in real-world situations [54]. Lastly, model-based RL algorithms have the capacity to predict unexpected actions and states, which also provides a more controlled learning process.

In summary, model-free RL algorithms learn through exploration, whereas model-based RL algorithms learn by simulating scenarios [41]. The slight difference between model-based RL and model-free is shown in Fig. 4. In addition, the summary of the distinctions between Model-Based and Model-Free Reinforcement Learning (RL) algorithms is also shown in Table I.

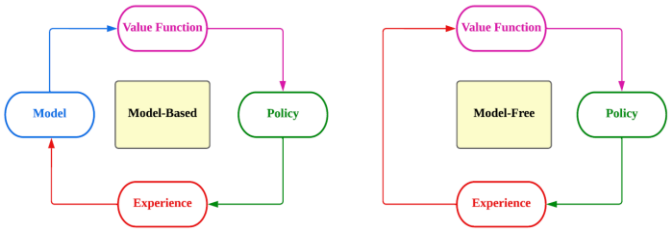

Fig. 4. Difference between model-based and model-free RL.

TABLE I. SUMMARY OF THE DISTINCTIONS BETWEEN MODEL-BASED AND MODEL-FREE RL ALGORITHMS

Category	Model-Based	Model-Free
Learning Type	Indirect method	Direct method
Objective	Learns value, and policy functions and constructs an explicit model of the system	Learns policy or value function without explicit model construction
Data Efficiency	Outperforms in terms of data efficiency	May demand substantial data for satisfactory performance
Implementation	Challenging in real-world scenarios due to complexity and cost	Easier implementation, no explicit modelling required
Environment Interaction	Predict future states and rewards through explicit environment modelling	Direct interaction, enhances performance through exploration
Challenges	Complex model construction may introduce bias for learned models	Hard to learn complex problems in real-world scenarios
Applicability to Real World	Balancing model accuracy and real-world complexity is a significant challenge	Hard to implement due to time and cost constraints

B. Model-based: Given the Model and Learn the Model

The algorithms that use models are called model-based methods. In model-based Reinforcement Learning (RL), given-the-model and learn-the-model are two main types of approaches [48]. One of the examples of the given-the-model is the AlphaGo algorithm [16]. In this algorithm, AlphaGo is explicitly learned the rules of the board game and can be described using coding or computer language. Then, the transitions and rewards are known to the agent, which allows for the evaluation of different strategies through trial and error to get optimal results and iteratively improve the policy. This approach shows that a pre-determined model of the environment to guide the learning process and enhance the decision-making. For another example, the Monte Carlo Tree Search (MCTS) algorithm can be using a given model to simulate possible future states and evaluate action sequences for optimal planning [55].

Moreover, an example of the "learn the model" category in model-based Reinforcement Learning is the World Models algorithm [56]. One of the examples for learn the model approach is DreamerV2 [57]. DreamerV2 builds an internal world model of the environment by learning from its experiences, which can allow the agent to simulate trajectories in its "imagination" rather than only rely on real-world interactions. This can cause the agent to explore and learn optimal policies more efficiently, as it can try out different actions and observe hypothetical outcomes within its model, thus significantly reducing the need for real-world samples. Another example is Probabilistic Ensembles with Trajectory Sampling (PETS), which use the models to predict possible future states with uncertainties included [58]. PETS uses these learned dynamics to perform planning and action selection then helping the agent to handle uncertainty and make more robust decisions in those unpredictable environments. This approach allows the agent to improve sample efficiency by using imagined rollouts for planning while adapting to changes in real-world scenarios. The comparison of “Given-the-Model" and "Learn-the-Model" in model-based RL is shown in Table II.

TABLE II. COMPARISON BETWEEN "GIVEN THE MODEL" AND "LEARN THE MODEL"

Category	Given the Model	Learn the Model
Learning Type	Uses an explicitly specified model of the environment	Learns a model of the environment from gathered data
Decision-Making	Enhances decision-making through trial and error	Optimizes policies by leveraging insights from the model
Advantages	1. Allows for the evaluation of different strategies 2. Facilitates iterative improvement of the policy	1. Adapts to complex and unknown environments 2. Can generalize to various scenarios
Disadvantages	1. Require explicit specification of the environment 2. Limited adaptability with unforeseen changes	1. Require extensive data for accurate model learning 2. Complexity in training and interpreting the learned model

C. Model-Free: On-Policy and Off-Policy

Model-free Reinforcement Learning (RL) is typically categorized into on-policy and off-policy approaches [48, 41]. The on-policy approach strives to enhance and learn through the policy itself which is used for decision-making [59]. For the on-policy approach, the agent itself interacts with the environment, and the policy to interact with the environment and the improved policy remain the same. According to Singh et al.'s (2000) study, this policy algorithm could be more stringent because the updating of the value function is contingent on the experiences gained from implementing the policy [60].

On the other hand, the off-policy approach aims to improve a policy that is different from the one that is used to generate the data [48]. Unlike the on-policy approach, the off-policy approach does not require the same agent that interacts with the environment. The experiences of other agents interacting with the environment can also be utilized to enhance the policy. When the agent learns the behavior in one way is called the target policy, while when it is learned using data generated by another policy is known as the behavior policy [61]. In addition, the agent learns from data generated by a behavior policy that might be explored more widely than the target policy, which allows for more efficient learning. This flexibility allows for more diverse data sources to contribute to

the policy improvement process. In Fakoor et al. (2020) research, it is noted that off-policy methods encounter bias issues as the data from an outdated policy differs from the current policy, making it unsuitable to update the current policy's value function using old data [62]. On the other hand, on-policy methods avoid bias but may face variance challenges, tending to be more data-efficient as they focus on the current samples.

One of the examples of an on-policy approach is SARSA State-Action-Reward-State-Action (SARSA) [41, 48]. In the SARSA algorithm, an action is selected based on the current policy and executed. Then, the data is utilized to update the current policy. In the on-policy setting, the policy that interacts with the environment is the same as the updated policy, which ensures consistency between the policy used during interaction and the one that is improved. The SARSA update function is shown below:

$$Q\{S(t),A(t)\} \leftarrow Q\{S(t),A(t)\} + \alpha[Q\{S(t+1),A(t+1)\} - Q\{S(t),A(t)\} \quad (1)$$

In this equation,

- $Q\{S(t),A(t)\}$ represents the Q-value for the state action pair at time t
- α is the learning rate
- $Q\{S(t+1),A(t+1)\}$ is the Q value for the next state action pair at time $t+1$

This updated function shows how SARSA adjusts the Q-values based on the observed rewards and transitions, which also continues to refine the policy in an on-policy manner. In the off-policy category, one of the examples is Q-learning, which employs the max operation and a greedy policy when selecting actions [41, 43, 48]. In addition, Q-learning involves updating a policy that interacts with the environment and the updated policy which is not necessarily the same as the policy used during interaction.

The Q-learning update function is shown below:

$$Q\{S(t),A(t)\} \leftarrow Q\{S(t),A(t)\} + \alpha[R(t+1) + \gamma \max_{\alpha} Q\{S(t+1),A(t+1)\} - Q\{S(t),A(t)\} \quad (2)$$

In this equation,

- $Q\{S(t),A(t)\}$ represents the Q-value for the state action pair at time t
- α is the learning rate
- $R(t+1)$ is the reward at time $t+1$
- γ is the discount reward
- $\max_{\alpha} Q\{S(t+1),A(t+1)\}$ is the maximum Q value for the next state S(t+1)

The function above also reflects how Q-learning iteratively refines the Q values based on the observed rewards and transitions and improves the policy over time. The comparison between "On-Policy" and "Off-Policy" in model-free RL is shown in Table III.

TABLE III. SUMMARY OF THE COMPARISON BETWEEN "ON-POLICY" AND "OFF-POLICY"

Category	*Given the Model*	*Learn the Model*
Learning Type	The agent learns through the policy used for decision-making.	Aims to improve a policy different from the data-generating policy.
Environment Interaction	The agent interacts with the environment using the policy.	Doesn't require the same agent to interact and data from other agents can be used (shared).
Consistency	The policy used during interaction and improved policy are the same.	Involves a target policy (learned) and a behavior policy (data-generating).
Flexibility	Limited by the exploration of the current policy.	Learns from data generated by a behavior policy that may be explored more widely.

D. Value-based Approach and Policy-based Approach

The value-based approach typically involves learning the value function through methods such as Temporal difference (TD) learning, Q-Learning, or Deep Q-Network (DQN) [63, 64, 65, 66]. This technique aims to identify the optimal action to take and the action under this approach tends to be deterministic, such that they are chosen with a clear understanding of consequences. The value function operates by working backward from the target state and attributing rewards to the preceding state. This approach can be helped in the selection of only one action that leads towards achieving the desired outcome and closer to the goal [65]. In summary, it involves a strategic evaluation of the value of actions to make informed decisions and optimize the learning process.

In contrast to the value-based approach, the policy-based approach focuses on learning the conditional probability π of a policy through techniques such as the policy gradient method [67]. Instead of obtaining a value function like mentioned above, this approach directly determines the policy. Due to the stochastic action probability, the policy-based approach is more suitable for the application with large and continuous action [65]. At the same time, action selection becomes probabilistic with actions chosen based on their likelihood of efficiently reaching the desired outcome as dictated by the learned policy.

IV. THE ADVANCEMENT OF REINFORCEMENT LEARNING APPROACHES

In recent years, Reinforcement Learning (RL) has been improving with a fast pace and developed advanced approaches for increasingly complex problems in the real world. This review would like to focus on Deep Reinforcement Learning, Hierarchical Reinforcement Learning, Multi-Agent Reinforcement Learning and Hybrid Model Based Reinforcement Learning. These approaches have expanded the range of high dimensional RL applications, multi-agent applications, hierarchical decision-making applications and optimal policies or strategies by a combination of model based and model free methods.

A. Deep Reinforcement Learning (DRL)

Deep Reinforcement Learning (DRL) combines Reinforcement Learning (RL) and deep learning to enable agents to learn optimal policies for decision-making tasks through trial and error [14, 15, 68, 69]. The DRL is known for utilizing the principle of RL with the theory of deep learning to facilitating those automatic extractions of features from the input and benefits for in the fields such as robotic, autonomous driving and video games [14, 70, 71].

One of the achievements in DRL was the development of Deep Q Network (DQN) by work of Mnih et al. (2015), which shows the human level performance on Atari games using raw pixel inputs. The DQN uses the convolutional network to approximate the Q value function, which trained using variant of Q-learning with experience and target network to stabilize training [4]. Subsequently, Schulman et al. (2015) have introduced the Trust Regio Policy Optimization (TRPO), which addressing the not stable and inefficiency of policy gradient methods. TRPO ensures monotonic improvement by optimizing objective function subject to a trust region constraint and make it more stable and reliable training [13, 72]. Further refinement came with the Proximal Policy Optimization (PPO) work by Schuman et al. (2017), which simplified the algorithm and enhanced computational efficiency by using clipped objective to balance exploration and exploitation effectively [73]. Lillicrap et al. (2015) have also extended the actor-critic framework to continuous action spaces with the Deep Deterministic Policy Gradient (DDPG) algorithm. DDPG will employ the actor network to parametrize the policy and network to estimate the Q- value function which enabling the application of DRL such as robotic control task [74].

In addition, Kostrikov et al. (2021) proposed the Implicit Q-learning (IQL) algorithm, which is an offline Reinforcement Learning method that avoids evaluating unseen actions, thereby mitigating errors from distributional shift. By leveraging state-value functions as random variables and conditionally using the expected value of the state, IQL can improve the policy without directly querying actions from the distribution [75]. On other hand, Chen et al. (2022) have also proposed the DreamerV2 algorithm, which builds on the Dreamer framework by incorporating discrete latent variables and advanced world model. It is also able to demonstrate a similar performance on Atari benchmark with efficient performance [76]. Sekar et al. (2020) introduced the Plan2Explore algorithm, which emphasizes intrinsic exploration by using a self-supervised world model to plan for expected future novelty, enabling the agent to efficiently explore and quickly adapt to multiple downstream tasks [77].

In summary, the advancement of DRL has revolutionized the fields of RL by enabling the agents to learn from high dimensional inputs to perform complex tasks. The new algorithms such as DQN, TRPO, PPO, DDPG, IQL, DreamerV2 and Plan2Explore have advanced the application of RL in different fields including gaming, robotics and autonomous systems. As research in DRL continues to focus, the efficiency, stability and generalization of RL will have further improvement. Fig. 5 provides a schematic illustration of DQN and Plan2Explore, presented as a case study in DRL.

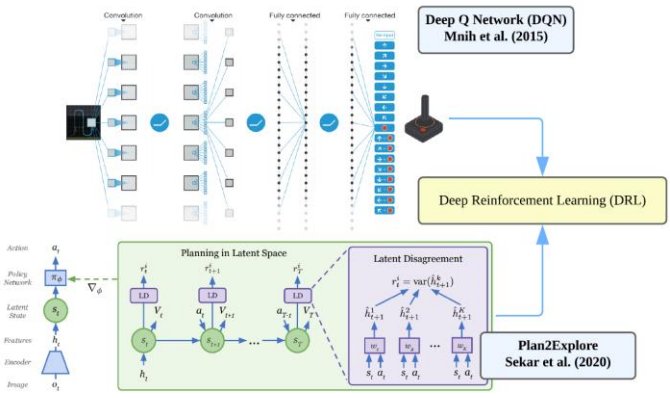

Fig. 5. Schematic illustration of Deep Q-Network (DQN) and Plan2Explore. [4, 77].

B. Hierarchical Reinforcement Learning (HRL)

Hierarchical Reinforcement Learning (HRL) is one of the approaches in Reinforcement Learning fields that able to addresses the challenges faced by traditional Reinforcement Learning method which include scalability and efficiency with the tasks that required long term planning [37, 78, 79]. HRL is able to solve these problems by decomposing them into hierarchy of subs tasks [37, 80]. The core idea of HRL can be described as hierarchical structure where higher level policies select sub tasks and lower level policies execute actions to achieve the goal of subs task.

One of the foundational works in HRL is Feudal Reinforcement Learning (FRL) by Dayan and Hinton (1992) which introduced a hierarchical structure where higher level manager set goals for lower level workers [81]. Each hierarchical level operates in different temporal and spatial resolution, allowing the agent to decompose complex tasks into simpler sub-tasks. In addition, another early work in HRL is the Options framework introduced by Sutton et al. (1999), this framework introduces the concept of options, which temporarily extended actions that consists of policy, termination conditions and initiation set [82]. These options can allow the agents to operate at different time scales and make the learning more efficient. Moreover, the more recent advancements in HRL include the Hierarchical-DQN (h-DQN) framework, which extends the DQN algorithm by incorporating hierarchical structure. The h-DQN framework consists of meta-controller that selects sub-goals and lower level controller which learns to achieve the sub goals using DQN, be able to apply to Atari games and navigating 3D environments [83]. The Options-Critic architecture by Bacon et al. (2017) provides a framework for learning both options and policies over options in end-to-end manner, the architecture introduces intra-option policy gradient methods to optimize the policies within options and termination conditions [84].

Furthermore, Vezhnevets et al. (2016) proposed the Strategic Attentive Writer (STRAW) framework, which is a deep recurrent neural network (RNN) architecture that capable learning macro-actions in a Reinforcement Learning setting. The model builds an internal plan and partitions it into sub-sequences then allowing the agent to commit to a plan for a period before replanning. This approach allowed the agent to

explore and compute efficiently across different tasks [85]. The Hierarchical Reinforcement Learning with Off-policy correction (HIRO) algorithm introduced by Nachum et al. (2018) addresses the challenges of non-stationarity in HRL by introducing an off-policy correction mechanism, enabling stable and efficient learning of hierarchical policies [79]. Levy et al. (2019) introduced the Hierarchical Actor-Critic (HAC) algorithm, which extends the actor-critic framework to a hierarchical setting by enabling agents to operate at multiple levels of abstraction simultaneously [86].

Recent developments after 2020 in HRL have further expanded, The Hierarchical Variational Autoencoder (HVAE) framework introduced by Bai et al. (2023) combines probabilistic generative models with deep neural networks to learn hierarchical topic representations for multi-view text documents. HVAE captures both local and global topical information, enabling efficient modelling of complex document structures [87]. The Hierarchical Deep Reinforcement Learning with Automatic Sub-Goal Identification via Computer Vision (HADS) by Liu et al. (2021) introduces a sub-goal generation mechanism that adapts to the agent's learning progress, applied to tasks such as game manipulation and navigation [88]. The Hierarchical Deep Reinforcement Learning with Graph Neural Networks (HRLOrch) introduced by Li & Zhu (2021) uses graph neural networks to model the hierarchical structure of the environment at multiple levels of abstraction [89].

In summary, HRL has significantly advanced the RL field by mainly enabling the agents to decompose complex tasks to simpler sub-tasks, hence cause the learning and planning more efficiently. The development of HRL frameworks including Options framework, FRL, h-DQN, Option-Critic, STRAW, HIRO, HAC, HVAE, HADS and HRLOrch, which further improves in terms of efficiency, stability and generalization capabilities. The schematic illustration of h-DQN, STRAW and HRLOrch is shown in Fig. 6.

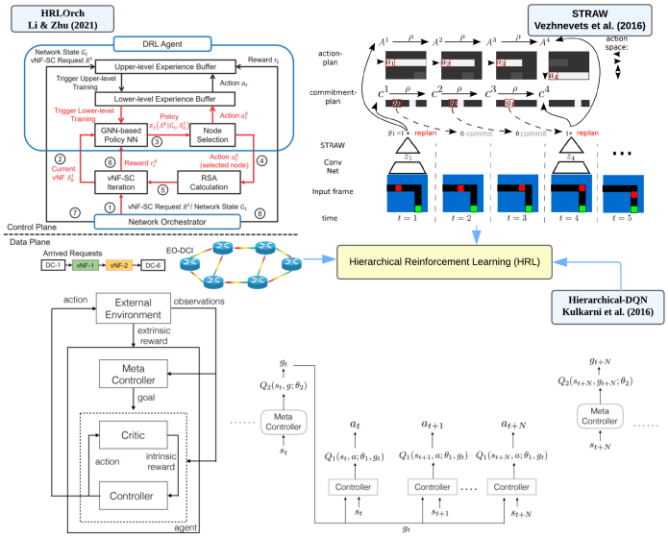

Fig. 6. Schematic illustration of h-DQN, STRAW and HRLOrch [83, 85, 89].

C. Multi-Agent Reinforcement Learning (MARL)

Multi-Agent Reinforcement Learning (MARL) is one of the specialized areas in Reinforcement Learning (RL) that focuses on the environment where multiple agents interact (not limited to number of environments), each aiming to optimize the performance while considering the presence and actions of other agents [18, 90, 91, 92]. Unlike single agent, each agent in MARL has its own goal, which may involve cooperation, competition or a mix of both, hence the environment is non-stationary from the perspective of each agent because other agents are also learning and changing their policies [93, 94].

One of the significant advancements in MARL is the Differentiable Inter-Agent Learning (DIAL) algorithm by Foerster et al. (2016), which uses differentiable communication channels to enable end-to-end training of communication policies, allowing agents to learn to communicate more effectively [95]. In addition, Lowe et al. (2017) introduced the Multi-Agent Deep Deterministic Policy Gradient (MADDPG) algorithm, which uses centralized training to gather global information while employing decentralized execution for deployment. Each agent has its own policy and critic, but the critics have access to the global state and actions of all other agents during training, making the training process more stable and effective [96].

For more recent advancements in MARL, Iqbal and Sha (2019) introduced the Actor-Attention-Critic (AAC) framework, which uses an attention mechanism to focus on relevant parts of the environment and other agents' actions [97]. This framework enhances the scalability and performance of MARL algorithms in more complex environments. In addition, Rashid et al., 2020 proposed the QMIX algorithm, which decomposes the joint action-value function into a monotonic combination of individual value functions for agents, enabling efficient coordination in cooperative tasks [98]. Yu et al. (2022) introduced Multi-Agent Proximal Policy Optimization (MAPPO), an extension of the PPO algorithm for multi-agent settings. MAPPO employs centralized training with decentralized execution to address cooperative and complex tasks, achieving performance comparable to off-policy methods like MADDPG [99].

Furthermore, Carta et al. (2021) proposed an ensemble approach using multiple Deep Q-learning (Multi-DQN) agents to enhance stock market forecasting by training several agents on the same data and aggregating their decisions [100]. Zhang et al. (2022) introduced the Multi-Agent Graph Convolutional Reinforcement Learning (MAGC) framework, which employs graph neural networks to model the interactions between agents. MAGC enables agents to learn and coordinate their actions more effectively by capturing the relational structures of the environment, and it is applied to tasks such as dynamic electric vehicle charging pricing [36]. In summary, the development of MARL frameworks including DIAL, MADDPG, AAC, QMIX, MAPPO, Multi-DQN and MAGC further advanced the field of RL through the multi agent systems. Examples of schematic illustrations of Multi-DQN and MADDPG are shown in Fig. 7.

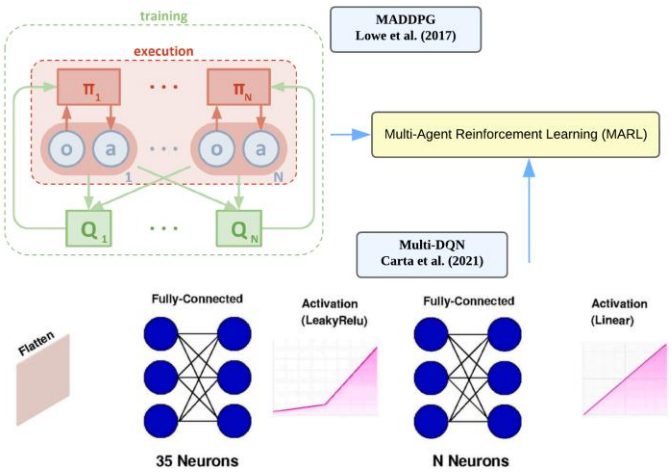

Fig. 7. Schematic illustration of multi-DQN and MADDPG [96, 100].

D. Model-Based Reinforcement Learning (MBRL)

Model-Based Reinforcement Learning (MBRL) is one of the approaches in Reinforcement Learning (RL) field that focuses on building models of environment to improve learning and decision-making effectiveness for gaming and robotic tasks [49, 101, 102]. The difference between model-based and model-free was discussed in previous section Model-Based and Model-Free. The core idea of MBRL is mainly decompose the objective into two main component which are model learning and planning, model learning includes environment's transition dynamics and reward function while planning uses the learned model to simulate future reward [102].

The Dyna framework that introduced by Sutton (1991) is one of the earliest works in MBRL, which integrates model learning and planning by combined real world interactions and simulated experiences generated by learned model, it allows the agent to update policy using real and synthetic data [103]. In addition, Delsenroth & Rasmussen (2011) have proposed the Probabilistic Inference for Learning Control (PILCO) algorithm which is one type of model-based approach that uses Gaussian processes to model the environment's dynamics [104]. Nagabandi et al. (2018) introduced a model-based RL approach using neural network dynamics (MBRL-NN). This method employs deep neural networks to model the environment's transitions and combines them with model predictive control (MPC) for planning [105].

Furthermore, Ha and Schmidhuber (2018) proposed the World Models framework, which learns a compact, latent representation of the environment using a Variational Autoencoder (VAE) and a Recurrent Neural Network (RNN). The agent plans and acts within this learned model, achieving good results on tasks in CarRacing-v0 and VizDoom [56]. The Probabilistic Ensembles with Trajectory Sampling (PETS) algorithm addresses model uncertainty in Model-based Reinforcement Learning (MBRL) by using an ensemble of probabilistic neural networks to model environment dynamics and trajectory sampling to account for uncertainty [58]. Moreover, Janner et al. (2019) introduced Model-Based Policy Optimization (MBPO), which integrates model-based and model-free approaches to improve sample efficiency. MBPO utilizes an ensemble of probabilistic neural networks to model the environment dynamics and conducts policy optimization within this learned model [106].

In recent years, Schrittwieser et al. (2020) proposed MuZero, a model-based RL algorithm that learns a model of the environment's dynamics and uses it for planning without requiring prior knowledge of the environment. MuZero achieves state-of-the-art performance in Atari, Go, chess, and shogi, demonstrating the benefits of combining model-based planning with model-free learning [107]. Similar with Deep Reinforcement Learning (DRL), the DreamerV2, MuZero and Plan2Explore algorithm also consider advancement for MBRL which expanded the application of MBRL to wider range of field. In summary, the introduction of Dyna, PILCO, MBRL-NN, World Models, PETS, MBPO and MuZero framework have advanced the field of RL and improved in terms of learning and decision-making. Schematic illustration of PETS and MuZero are shown in Fig. 8, which shows how the model plans and communicates with environments. The summary and comparative analysis of advanced Reinforcement Learning (RL) approaches is shown in Table IV.

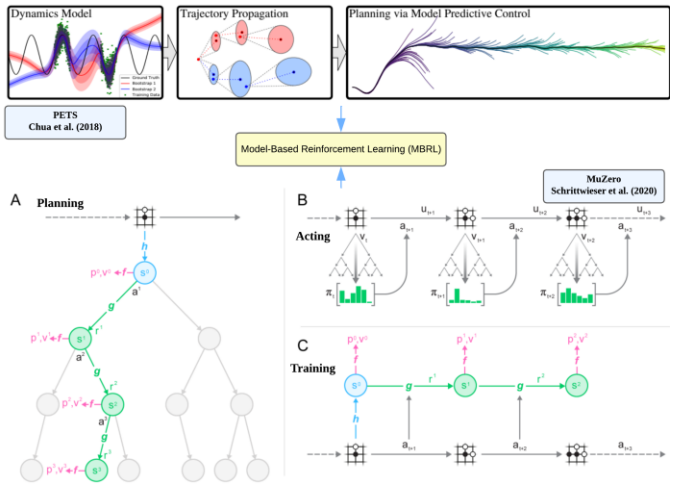

Fig. 8. Schematic illustration of PETS and MuZero [58, 107].

TABLE IV. SUMMARY AND COMPARATIVE ANALYSIS OF ADVANCED REINFORCEMENT LEARNING (RL) APPROACHES

Category	Framework	Key Contributions	Author & Year
Deep Reinforcement Learning (DRL)	Deep Q Network (DQN)	Approximates Q-value function using CNN, stabilizes training with experience replay and target networks.	Mnih et al., 2015
	Trust Region Policy Optimization (TRPO)	Ensures stable training via trust region constraints.	Schulman et al., 2015
	Proximal Policy Optimization (PPO)	Simplified policy gradient method with a clipped objective for efficiency.	Schulman et al., 2017

Category	Framework	Key Contributions	Author & Year
	Deep Deterministic Policy Gradient (DDPG)	Extends actor-critic framework to continuous action spaces.	Lillicrap et al., 2015
	Implicit Q-Learning (IQL)	Mitigates distributional shift errors in offline RL by using state-value functions as random variables.	Kostrikov et al., 2021
	DreamerV2	Combines latent variables with world models for efficient decision-making.	Chen et al., 2022
	Plan2Explore	Intrinsic exploration using a self-supervised world model.	Sekar et al., 2020
Hierarchical Reinforcement Learning (HRL)	Feudal RL (FRL)	Hierarchical decomposition of tasks via manager-worker relationships.	Dayan & Hinton, 1992
	Options Framework	Temporally extended actions with initiation, termination, and policies.	Sutton et al., 1999
	Hierarchical DQN (h-DQN)	Meta-controller for sub-goals and DQN-based controller.	Kulkarni et al., 2016
	Option-Critic Architecture	End-to-end learning of options and intra-option policies.	Bacon et al., 2017
	Strategic Attentive Writer (STRAW)	Plans macro actions via deep RNNs for efficient exploration and computation.	Vezhnevets et al., 2016
	Hierarchical Reinforcement learning with Off-policy correction (HIRO)	Off-policy correction for stable hierarchical learning.	Nachum et al., 2018
	Hierarchical Actor-Critic (HAC)	Actor-critic framework for multi-level task abstraction.	Levy et al., 2019
	Hierarchical Deep Reinforcement Learning with Automatic Sub-Goal Identification via Computer Vision (HADS)	Sub-goal generation via computer vision for dynamic task adaptation.	Liu et al., 2021
	Hierarchical Deep Reinforcement Learning with Graph Neural Networks (HRLOrch)	Graph neural networks for multi-level environment abstraction.	Li & Zhu, 2021
	Hierarchical Variational Autoencoder (HVAE)	Combines probabilistic generative models with deep neural networks to learn hierarchical topic representation	Bai et al., 2023
Multi-Agent Reinforcement Learning (MARL)	Differentiable Inter-Agent Learning (DIAL)	End-to-end learning of communication policies via differentiable channels.	Foerster et al., 2016
	Multi-Agent Deep Deterministic Policy Gradient (MADDPG)	Centralized training with decentralized execution for multi-agent setups.	Lowe et al., 2017
	Actor-Attention-Critic (AAC)	Attention mechanism for focusing on relevant environment and agent interactions.	Iqbal & Sha, 2019
	QMIX	Monotonic decomposition of joint action-value for cooperative tasks.	Rashid et al., 2020
	Multi-Agent Proximal Policy Optimization (MAPPO)	Extends PPO for multi-agent systems with centralized training and decentralized execution.	Yu et al., 2022
	Multiple Deep Q-learning (Multi-DQN)	Ensemble of DQN agents for aggregating decisions in forecasting tasks.	Carta et al., 2021
	Multi-Agent Graph Convolutional Reinforcement Learning (MAGC)	Graph neural networks for relational multi-agent modelling.	Zhang et al., 2022
Model-Based Reinforcement Learning (MBRL)	Dyna	Combines model learning and planning using real and synthetic data.	Sutton, 1991
	Probabilistic Inference for Learning Control (PILCO)	Uses Gaussian processes to model environment dynamics.	Delsenroth & Rasmussen, 2011
	Model-Based Reinforcement Learning using Neural Network Dynamics (MBRL-NN)	Combines neural network-based dynamics with model predictive control.	Nagabandi et al., 2018
	World Models	Latent environment representation via VAE and RNN.	Ha & Schmidhuber, 2018
	Probabilistic Ensembles with Trajectory Sampling (PETS)	Ensemble probabilistic models with trajectory sampling for uncertainty handling.	Chua et al., 2018
	Model-Based Policy Optimization (MBPO)	Combines model-based and model-free approaches for sample efficiency.	Janner et al., 2019
	MuZero	Learns environment models and plans without prior knowledge, achieving good results in gaming.	Schrittwieser et al., 2020

V. Challenges and Alternative Solutions

Although Reinforcement Learning (RL) has its effectiveness in a wide range of applications, it still faces significant challenges that limit the efficiency, scalability, and real-world applicability. The main key challenges include sample inefficiency, exploration-exploitation dilemma, and difficulties with generalization across different tasks and environments. Hence, there is much research that proposed new approaches to improve the adaptability and efficiency of

RL agents such as curiosity-driven exploration, meta-learning, and transfer learning.

One of the important challenges in RL is sample inefficiency as mentioned, where agents require large amount of interaction with the environment to learn effective policies. For example, in complex tasks such as involving high dimensional state space or continuous actions space, RL methods usually required more time to converge to optimal solutions. For the sample inefficiency challenges, several solutions have been proposed. For instance, the work of Janner et al. (2019) have proposed an RL algorithm Model-Based Policy Optimization (MBPO) which uses short model-generated rollouts to improve sample efficiency and performance as mentioned [106]. In addition, Soft Actor-Critic (SAC) which an off-policy actor-critic algorithm based on maximum entropy RL can be used to maximize both expected reward and entropy, it also able to be enabling agents to learn from data collected under different policies [108]. Not only that, Deep Q-Networks (DQN) also can be used to store and reuse past experiences or learning, this can reduce the need for time required and samples in each iteration [13].

In addition, one of another challenges in RL are exploration-exploitation dilemma, in which an agent must balance between exploring new environment then found the highest beneficial actions and exploiting known actions that produce high rewards. Poor exploration strategies can lead to suboptimal strategies, especially in environments where reward signals are delayed or require a long time such as in physical design. Therefore, several solutions have been proposed for solving this issue. One of the solutions is curiosity-driven exploration which to explores using curiosity as an intrinsic reward signal for agents in environments with sparse or no extrinsic rewards which curiosity is defined as the error in predicting the consequences of the agent's actions in a visual feature space [109]. In addition, curiosity-driven exploration is important for autonomous learning, highlighting various algorithmic models that capture different aspects of this process [110]. Moreover, entropy regularization also can be used to address the exploration-exploitation dilemma by introducing f-divergence penalties [111]. These penalties ensure that the policy does not deviate too much from the current policy, promoting balanced exploration and exploitation. By adjusting the divergence function, the agent can control the trade-off between exploring new actions and exploiting known rewarding actions, this can lead to more stable and efficient learning dynamics.

Furthermore, another challenge is difficulties with generalization, where generalization in RL refers to the ability of an agent to still perform well in new or unseen environments [112, 113]. This is due to RL models often overfitting to specific environments during training, leading to a decline in performance when faced with new or unseen scenarios. This issue is a serious problem for real-world applications such as autonomous driving, where the agent is required to handle various scenarios under different conditions [115]. In order to solve this problem, the model agnostic meta learning (MAML) approach can be applied to enable agents more quickly adapt to new tasks by learning from distribution of related tasks during the training phase [115, 116, 117]. This approach can make the agent more robust and adaptable in unfamiliar environments. In addition, transfer learning also can be applied to transfer the knowledge to another related task to reducing the training data needed in new environment [118, 119] This will be more useful when the training in the target environment is costly such as required more memory space. Moreover, domain randomization also can help to improve the generalization issue by training the agents in varying the parameters, so that the agents can be handle the real-world variability more effectively and more adaptable to new, unseen environments [120].

In summary, the purpose of artificial intelligence (AI) including RL is not to replace humans, AI is designed to enhance human efficiency and achieve better outcomes. Humans are required to treat it as a tool to increase efficiency, streamline workflows, and assist in decision-making processes. However, it is also very important to ensure that AI technologies are applied properly to maximize the potential benefits and minimize the risk of misuse or unintended consequences. The summary of alternative methods and their potential benefits is shown in Table V.

TABLE V. SUMMARY OF ALTERNATIVE METHODS AND POTENTIAL BENEFITS

Alternative Method	Challenges	Key Feature	Potential Benefits
Model-Based Policy Optimization (MBPO)	Sample Efficiency	Use short model-generated rollouts	Increases sample efficiency
Soft Actor-Critic (SAC)	Sample Efficiency	Agents can learn from data collected under different policies	Maximize both expected reward and entropy and purpose to increase sample efficiency
Curiosity-Driven Exploration	Exploration-Exploitation Dilemma	Use intrinsic rewards based on novelty	Enhances the exploration in sparse reward environments
Entropy Regularization	Exploration-Exploitation Dilemma	Maintains randomness in policy by penalizing determinism	Promotes continued exploration during training
Meta-Learning	Generalization	Learns to adapt quickly to new tasks from experiences	Improves adaptability and efficiency across tasks
Transfer Learning	Generalization	Transfers knowledge from one task to another task	Reduces training time and data requirements
Domain Randomization	Generalization	Trains in a different of simulated environments	Improves robustness to different of environments

VI. Conclusion and Outlook

In conclusion, the evolutions of Reinforcement Learning has successfully impacted the different fields from robotics and autonomous driving system, healthcare and finance. The integration of RL with different approach can further advanced the application, for example the integration of deep learning and RL which known as Deep Reinforcement Learning (DRL) has improved in solving the high dimensional problem and applied in AlphaGo.

In addition, the focus of RL is mainly improving the sample efficiency, safety and scalability for real world applications. The innovations in hierarchical Reinforcement Learning, transfer learning and domain randomization are expected to improve the adaptability and generalizability of RL systems. In summary, as increasing more effort in improving RL, it will play an important role in artificial intelligence and autonomous technologies which will help humans for more complex challenges in daily life.

Acknowledgment

This research was funded by the Ministry of Higher Education (MoHE) Malaysia with the Fundamental Research Grant Scheme (FRGS) under grant number FRGS/1/2021/TK0/UKM/02/17, and by the National Research and Innovation Agency (BRIN) Indonesia under the Perjanjian Pendanaan Program Riset dan Inovasi untuk Indonesia Maju Gelombang 4, with grant numbers 20/IV/KS/11/2023 and 1181/PL4.7.4.2/PT/2023.

References

[1] Naeem, M., Rizvi, S. T. H., & Coronato, A. (2020). A gentle introduction to reinforcement learning and its application in different fields. IEEE access, 8, 209320-209344.

[2] Peng, J., Jury, E. C., Dönnes, P., & Ciurtin, C. (2021). Machine learning techniques for personalised medicine approaches in immune-mediated chronic inflammatory diseases: applications and challenges. Frontiers in pharmacology, 12, 720694.

[3] Nian, R., Liu, J., & Huang, B. (2020). A review on reinforcement learning: Introduction and applications in industrial process control. Computers & Chemical Engineering, 139, 106886.

[4] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. Nature, 518, 529-533.

[5] AlMahamid, F., & Grolinger, K. (2021).A 1:A 38 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE) (pp. 1-7). IEEE.

[6] Bogert, K., Lin, J. F. S., Doshi, P., & Kulic, D. (2016). Expectation-maximization for inverse reinforcement learning with hidden data. In Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems (pp. 1034-1042).

[7] Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey. Journal of artificial intelligence research, 4, 237-285.

[8] Kober, J., Bagnell, J. A., & Peters, J. (2013). Reinforcement learning in robotics: A survey. The International Journal of Robotics Research, 32(11), 1238-1274.

[9] Cheng, Y., Chen, C., Hu, X., Chen, K., Tang, Q., & Song, Y. (2021). Enhancing mixed traffic flow safety via connected and autonomous vehicle trajectory planning with a reinforcement learning approach. Journal of Advanced Transportation, 2021(1), 6117890.

[10] Haydari, A., & Yılmaz, Y. (2020). Deep reinforcement learning for intelligent transportation systems: A survey. IEEE Transactions on Intelligent Transportation Systems, 23(1), 11-32.

[11] Hofmann, S., Walter, M., Servadei, L., & Wille, R. (2024). Thinking Outside the Clock: Physical Design for Field-coupled Nanocomputing with Deep Reinforcement Learning. In 2024 25th International Symposium on Quality Electronic Design (ISQED) (pp. 1-8). IEEE.

[12] Lu, Y. C., Chan, W. T., Guo, D., Kundu, S., Khandelwal, V., & Lim, S. K. (2023). RL-CCD: Concurrent clock and data optimization using attention-based self-supervised reinforcement learning. In 2023 60th ACM/IEEE Design Automation Conference (DAC) (pp. 1-6). IEEE

[13] Li, S. E. (2023). Deep reinforcement learning. In Reinforcement learning for sequential decision and optimal control (pp. 365-402). Singapore: Springer Nature Singapore.

[14] Arulkumaran, K., Deisenroth, M. P., Brundage, M., & Bharath, A. A. (2017). Deep reinforcement learning: A brief survey. IEEE Signal Processing Magazine, 34(6), 26-38.

[15] François-Lavet, V., Henderson, P., Islam, R., Bellemare, M. G., & Pineau, J. (2018). An introduction to deep reinforcement learning. Foundations and Trends® in Machine Learning, 11(3-4), 219-354.

[16] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., & Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. Nature, 529(7587), 484-489.

[17] Sallab, A. E., Abdou, M., Perot, E., & Yogamani, S. (2017). Deep reinforcement learning framework for autonomous driving. arXiv preprint arXiv:1704.02532.

[18] Zhang, K., Yang, Z., & Başar, T. (2021). Multi-agent reinforcement learning: A selective overview of theories and algorithms. Handbook of reinforcement learning and control, 321-384.

[19] Ning, Z., & Xie, L. (2024). A survey on multi-agent reinforcement learning and its application. Journal of Automation and Intelligence.

[20] Thorndike, E. L. (1933). A proof of the law of effect. Science, 77(1989), 173-175.

[21] Skinner, B. F. (1950). Are theories of learning necessary?. Psychological review, 57(4), 193.

[22] Skinner, B. F. (1958). Reinforcement today. American Psychologist, 13(3), 94.

[23] Islam, M. H. (2015). Thorndike theory and it's application in learning. At-Ta'lim: Jurnal Pendidikan, 1(1), 37-47.

[24] Skinner, B. F. (1963). Operant behavior. American psychologist, 18(8), 503.

[25] Ab Azar, N., Shahmansoorian, A., & Davoudi, M. (2020). From inverse optimal control to inverse reinforcement learning: A historical review. Annual Reviews in Control, 50, 119-138.

[26] Bellman, R. (1957). Dynamic programming princeton university press. Princeton, NJ, 4-9.

[27] Bellman, R., & Kalaba, R. (1957). Dynamic programming and statistical communication theory. Proceedings of the National Academy of Sciences, 43(8), 749-751.

[28] Bellman, R., & Lee, E. S. (1978). Functional equations in dynamic programming. Aequationes mathematicae, 17(1), 1-18.

[29] Ding, Z., Huang, Y., Yuan, H., & Dong, H. (2020). Introduction to reinforcement learning. Deep reinforcement learning: fundamentals, research and applications, 47-123.

[30] Howard, R. A. (1960). Dynamic Programming and Markov Processes. MIT Press google schola, 2, 39-47.

[31] Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. Machine learning, 3, 9-44.

[32] Watkins, C. J. C. H. (1989). Learning from delayed rewards.

[33] Granter, S. R., Beck, A. H., & Papke Jr, D. J. (2017). AlphaGo, deep learning, and the future of the human microscopist. Archives of pathology & laboratory medicine, 141(5), 619-621.

[34] Dulac-Arnold, G., Levine, N., Mankowitz, D. J., Li, J., Paduraru, C., Gowal, S., & Hester, T. (2021). Challenges of real-world reinforcement

learning: definitions, benchmarks and analysis. Machine Learning, 110(9), 2419-2468.

[35] Chen, X., Qu, G., Tang, Y., Low, S., & Li, N. (2022). Reinforcement learning for selective key applications in power systems: Recent advances and future challenges. IEEE Transactions on Smart Grid, 13(4), 2935-2958.

[36] Zhang, W., Liu, H., Han, J., Ge, Y., & Xiong, H. (2022). Multi-agent graph convolutional reinforcement learning for dynamic electric vehicle charging pricing. In Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining (pp. 2471-2481).

[37] Pateria, S., Subagdja, B., Tan, A. H., & Quek, C. (2021). Hierarchical reinforcement learning: A comprehensive survey. ACM Computing Surveys (CSUR), 54(5), 1-35.

[38] Zhu, Z., Lin, K., Jain, A. K., & Zhou, J. (2023). Transfer learning in deep reinforcement learning: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence.

[39] Huang, Q. (2020). Model-based or model-free, a review of approaches in reinforcement learning. In 2020 International Conference on Computing and Data Science (CDS) (pp. 219-221). IEEE

[40] Gao, C., & Wang, D. (2023). Comparative study of model-based and model-free reinforcement learning control performance in HVAC systems. Journal of Building Engineering, 74, 106852.

[41] Alrebdi, N., Alrumiah, S., Almansour, A., & Rassam, M. (2022). Reinforcement Learning in Image Classification: A Review. In 2022 2nd International Conference on Computing and Information Technology (ICCIT) (pp. 79-86). IEEE.

[42] Sutton, R. S., Barto, A. G., & Williams, R. J. (1992). Reinforcement learning is direct adaptive optimal control. IEEE control systems magazine, 12(2), 19-22.

[43] Watkins, C. J., & Dayan, P. (1992). Q-learning. Machine learning, 8, 279-292.

[44] Fan, J., Wang, Z., Xie, Y., & Yang, Z. (2020). A theoretical analysis of deep Q-learning. In Learning for dynamics and control (pp. 486-489). PMLR.

[45] Atkeson, C. G., & Santamaria, J. C. (1997). A comparison of direct and model-based reinforcement learning. In Proceedings of international conference on robotics and automation (Vol. 4, pp. 3557-3564). IEEE.

[46] Ni, T., Eysenbach, B., & Salakhutdinov, R. (2021). Recurrent model-free rl can be a strong baseline for many pomdps. arXiv preprint arXiv:2110.05038.

[47] Dayan, P., & Berridge, K. C. (2014). Model-based and model-free Pavlovian reward learning: revaluation, revision, and revelation. Cognitive, Affective, & Behavioral Neuroscience, 14, 473-492.

[48] Zhang, H., & Yu, T. (2020). Taxonomy of reinforcement learning algorithms. Deep Reinforcement Learning: Fundamentals, Research and Applications, 125-133.

[49] Polydoros, A. S., & Nalpantidis, L. (2017). Survey of model-based reinforcement learning: Applications on robotics. Journal of Intelligent & Robotic Systems, 86(2), 153-173.

[50] Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.

[51] Plaat, A., Kosters, W., & Preuss, M. (2023). High-accuracy model-based reinforcement learning, a survey. Artificial Intelligence Review, 56(9), 9541-9573.

[52] Lambert, N., Amos, B., Yadan, O., & Calandra, R. (2020). Objective mismatch in model-based reinforcement learning. arXiv preprint arXiv:2002.04523.

[53] Rajeswaran, A., Mordatch, I., & Kumar, V. (2020). A game theoretic framework for model based reinforcement learning. In International conference on machine learning (pp. 7953-7963). PMLR.

[54] Plaat, A., Kosters, W., & Preuss, M. (2020). Deep model-based reinforcement learning for high-dimensional problems, a survey. arXiv preprint arXiv:2008.05598.

[55] Browne, C. B., Powley, E., Whitehouse, D., Lucas, S. M., Cowling, P. I., Rohlfshagen, P., Tavener, S., Perez, D., Samothrakis, S., & Colton, S. (2012). A Survey of Monte Carlo Tree Search Methods. IEEE Transactions on Computational Intelligence and AI in Games, 4(1), 1–43

[56] Ha, D., & Schmidhuber, J. (2018). World models. arXiv preprint arXiv:1803.10122.

[57] Hafner, D., Lillicrap, T., Norouzi, M., & Ba, J. (2020). Mastering atari with discrete world models. arXiv preprint arXiv:2010.02193.

[58] Chua, K., Calandra, R., McAllister, R., & Levine, S. (2018). Deep reinforcement learning in a handful of trials using probabilistic dynamics models. Advances in neural information processing systems, 31.

[59] Sutton, R. S., Mahmood, A. R., & White, M. (2016). An emphatic approach to the problem of off-policy temporal-difference learning. The Journal of Machine Learning Research, 17(1), 2603-2631.

[60] Singh, S., Jaakkola, T., Littman, M. L., & Szepesvári, C. (2000). Convergence results for single-step on-policy reinforcement-learning algorithms. Machine learning, 38, 287-308.

[61] Maei, H. R., Szepesvári, C., Bhatnagar, S., & Sutton, R. S. (2010). Toward off-policy learning control with function approximation. In ICML (Vol. 10, pp. 719-726).

[62] Fakoor, R., Chaudhari, P., & Smola, A. J. (2020). P3o: Policy-on policy-off policy optimization. In Uncertainty in Artificial Intelligence (pp. 1017-1027). PMLR.

[63] Sewak, M. (2019). Policy-based reinforcement learning approaches: Stochastic policy gradient and the REINFORCE algorithm. Deep Reinforcement Learning: Frontiers of Artificial Intelligence, 127-140.

[64] Li, X., Lv, Z., Wang, S., Wei, Z., & Wu, L. (2019). A reinforcement learning model based on temporal difference algorithm. IEEE Access, 7, 121922-121930.

[65] Rammohan, S., Yu, S., He, B., Hsiung, E., Rosen, E., Tellex, S., & Konidaris, G. (2021). Value-Based Reinforcement Learning for Continuous Control Robotic Manipulation in Multi-Task Sparse Reward Settings. arXiv preprint arXiv:2107.13356.

[66] Singh, S. P., & Sutton, R. S. (1996). Reinforcement learning with replacing eligibility traces. Machine learning, 22, 123-158.

[67] Peters, J. (2010). Policy gradient methods. Scholarpedia, 5(11), 3698.

[68] Li, Y. (2017). Deep Reinforcement Learning: An Overview. arXiv preprint arXiv:1701.07274.

[69] Mousavi, S. S., Schukat, M., & Howley, E. (2018). Deep reinforcement learning: an overview. In Proceedings of SAI Intelligent Systems Conference (IntelliSys) 2016: Volume 2 (pp. 426-440). Springer International Publishing.

[70] Nguyen, H., & La, H. (2019). Review of deep reinforcement learning for robot manipulation. In 2019 Third IEEE international conference on robotic computing (IRC) (pp. 590-595). IEEE.

[71] Kiran, B. R., Sobh, I., Talpaert, V., Mannion, P., Al Sallab, A. A., Yogamani, S., & Pérez, P. (2021). Deep reinforcement learning for autonomous driving: A survey. IEEE Transactions on Intelligent Transportation Systems, 23(6), 4909-4926.

[72] Schulman, J., Levine, S., Abbeel, P., Jordan, M. & Moritz, P. (2015). Trust Region Policy Optimization. arXiv preprint arXiv:1502.05477.

[73] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347.

[74] Lillicrap, T. P. (2015). Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971.

[75] Kostrikov, I., Nair, A., & Levine, S. (2021). Offline reinforcement learning with implicit q-learning. arXiv preprint arXiv:2110.06169.

[76] Chen, C., Wu, Y. F., Yoon, J., & Ahn, S. (2022). Transdreamer: Reinforcement learning with transformer world models. arXiv preprint arXiv:2202.09481

[77] Sekar, R., Rybkin, O., Daniilidis, K., Abbeel, P., Hafner, D., & Pathak, D. (2020). Planning to explore via self-supervised world models. In International conference on machine learning (pp. 8583-8592). PMLR.

[78] Barto, A. G., & Mahadevan, S. (2003). Recent advances in hierarchical reinforcement learning. Discrete event dynamic systems, 13, 341-379.

[79] Nachum, O., Gu, S. S., Lee, H., & Levine, S. (2018). Data-efficient hierarchical reinforcement learning. Advances in neural information processing systems, 31.

[80] Al-Emran, M. (2015). Hierarchical reinforcement learning: a survey. International journal of computing and digital systems, 4(02).

[81] Dayan, P., & Hinton, G. E. (1992). Feudal reinforcement learning. Advances in neural information processing systems, 5.

[82] Sutton, R. S., Precup, D., & Singh, S. (1999). Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. Artificial intelligence, 112(1-2), 181-211.

[83] Kulkarni, T. D., Narasimhan, K., Saeedi, A., & Tenenbaum, J. (2016). Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. Advances in neural information processing systems, 29.

[84] Bacon, P. L., Harb, J., & Precup, D. (2017). The option-critic architecture. In Proceedings of the AAAI conference on artificial intelligence (Vol. 31, No. 1).

[85] Vezhnevets, A. S., Osindero, S., Schaul, T., Heess, N., Jaderberg, M., Silver, D., & Kavukcuoglu, K. (2017). Feudal networks for hierarchical reinforcement learning. In International conference on machine learning (pp. 3540-3549). PMLR.

[86] Levy, A., Konidaris, G., Platt, R., & Saenko, K. (2017). Learning multi-level hierarchies with hindsight. arXiv preprint arXiv:1712.00948.

[87] Bai, R., Huang, R., Qin, Y., Chen, Y., & Lin, C. (2023). HVAE: A deep generative model via hierarchical variational auto-encoder for multi-view document modeling. Information Sciences, 623, 40-55.

[88] Liu, C., Zhu, F., Liu, Q., & Fu, Y. (2021). Hierarchical reinforcement learning with automatic sub-goal identification. IEEE/CAA journal of automatica sinica, 8(10), 1686-1696.

[89] Li, B., & Zhu, Z. (2022). GNN-based hierarchical deep reinforcement learning for NFV-oriented online resource orchestration in elastic optical DCIs. Journal of Lightwave Technology, 40(4), 935-946.

[90] Tan, M. (1993). Multi-agent reinforcement learning: Independent vs. cooperative agents. In Proceedings of the tenth international conference on machine learning (pp. 330-337).

[91] Buşoniu, L., Babuška, R., & De Schutter, B. (2010). Multi-agent reinforcement learning: An overview. Innovations in multi-agent systems and applications-1, 183-221.

[92] Canese, L., Cardarilli, G. C., Di Nunzio, L., Fazzolari, R., Giardino, D., Re, M., & Spanò, S. (2021). Multi-agent reinforcement learning: A review of challenges and applications. Applied Sciences, 11(11), 4948.

[93] Papoudakis, G., Christianos, F., Rahman, A., & Albrecht, S. V. (2019). Dealing with non-stationarity in multi-agent deep reinforcement learning. arXiv preprint arXiv:1906.04737.

[94] Hernandez-Leal, P., Kaisers, M., Baarslag, T., & De Cote, E. M. (2017). A survey of learning in multiagent environments: Dealing with non-stationarity. arXiv preprint arXiv:1707.09183.

[95] Foerster, J., Assael, I. A., De Freitas, N., & Whiteson, S. (2016). Learning to communicate with deep multi-agent reinforcement learning. Advances in neural information processing systems, 29.

[96] Lowe, R., Wu, Y. I., Tamar, A., Harb, J., Pieter Abbeel, O., & Mordatch, I. (2017). Multi-agent actor-critic for mixed cooperative-competitive environments. Advances in neural information processing systems, 30.

[97] Iqbal, S., & Sha, F. (2019). Actor-attention-critic for multi-agent reinforcement learning. In International conference on machine learning (pp. 2961-2970). PMLR.

[98] Rashid, T., Samvelyan, M., De Witt, C. S., Farquhar, G., Foerster, J., & Whiteson, S. (2020). Monotonic value function factorisation for deep multi-agent reinforcement learning. Journal of Machine Learning Research, 21(178), 1-51.

[99] Yu, C., Velu, A., Vinitsky, E., Gao, J., Wang, Y., Bayen, A., & Wu, Y. (2022). The surprising effectiveness of ppo in cooperative multi-agent games. Advances in Neural Information Processing Systems, 35, 24611-24624.

[100] Carta, S., Ferreira, A., Podda, A. S., Recupero, D. R., & Sanna, A. (2021). Multi-DQN: An ensemble of Deep Q-learning agents for stock market forecasting. Expert systems with applications, 164, 113820.

[101] Kaiser, L., Babaeizadeh, M., Miłos, P., Osiński, B., Campbell, R. H., Czechowski, K., Erhan, D., Finn, C., Kozakowski, P., Levine, S., Mohiuddin, A., Sepassi, R., Tucker, G., & Michalewski, H. (2019). Model-based reinforcement learning for atari. arXiv preprint arXiv:1903.00374.

[102] Moerland, T. M., Broekens, J., Plaat, A., & Jonker, C. M. (2023). Model-based reinforcement learning: A survey. Foundations and Trends® in Machine Learning, 16(1), 1-118.

[103] Sutton, R. S. (1991). Dyna, an integrated architecture for learning, planning, and reacting. ACM Sigart Bulletin, 2(4), 160-163.

[104] Deisenroth, M., & Rasmussen, C. E. (2011). PILCO: A model-based and data-efficient approach to policy search. In Proceedings of the 28th International Conference on machine learning (ICML-11) (pp. 465-472).

[105] Nagabandi, A., Kahn, G., Fearing, R. S., & Levine, S. (2018). Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In 2018 IEEE international conference on robotics and automation (ICRA) (pp. 7559-7566). IEEE.

[106] Janner, M., Fu, J., Zhang, M., & Levine, S. (2019). When to trust your model: Model-based policy optimization. Advances in neural information processing systems, 32.

[107] Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., Guez, A., Lockhart, E., Hassabis, D., Graepel, T., Lillicrap, T., & Silver, D. (2020). Mastering atari, go, chess and shogi by planning with a learned model. Nature, 588(7839), 604-609.

[108] Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In International conference on machine learning (pp. 1861-1870). PMLR.

[109] Pathak, D., Agrawal, P., Efros, A. A., & Darrell, T. (2017). Curiosity-driven exploration by self-supervised prediction. In International conference on machine learning (pp. 2778-2787). PMLR.

[110] Ten, A., Oudeyer, P. Y., & Moulin-Frier, C. (2022). Curiosity-driven exploration. The Drive for Knowledge: The Science of Human Information Seeking, 53.

[111] Belousov, B., & Peters, J. (2019). Entropic regularization of markov decision processes. Entropy, 21(7), 674.

[112] Wang, K., Kang, B., Shao, J., & Feng, J. (2020). Improving generalization in reinforcement learning with mixture regularization. Advances in Neural Information Processing Systems, 33, 7968-7978.

[113] Di Langosco, L. L., Koch, J., Sharkey, L. D., Pfau, J., & Krueger, D. (2022). Goal misgeneralization in deep reinforcement learning. In International Conference on Machine Learning (pp. 12004-12019). PMLR.

[114] Coelho, D., Oliveira, M., & Santos, V. (2023). RLAD: Reinforcement Learning From Pixels for Autonomous Driving in Urban Environments. IEEE Transactions on Automation Science and Engineering.

[115] Finn, C., Abbeel, P., & Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. In International conference on machine learning (pp. 1126-1135). PMLR.

[116] Gurumurthy, S., Kumar, S., & Sycara, K. (2020). Mame: Model-agnostic meta-exploration. In Conference on Robot Learning (pp. 910-922). PMLR.

[117] Baik, S., Hong, S., & Lee, K. M. (2020). Learning to forget for meta-learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 2379-2387).

[118] Pan, S. J., & Yang, Q. (2009). A survey on transfer learning. IEEE Transactions on knowledge and data engineering, 22(10), 1345-1359.

[119] Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., & He, Q. (2020). A comprehensive survey on transfer learning. Proceedings of the IEEE, 109(1), 43-76.

[120] Slaoui, R. B., Clements, W. R., Foerster, J. N., & Toth, S. (2019). Robust domain randomization for reinforcement learning.