REINFORCEMENT LEARNING - A SYSTEMATIC LITERATURE REVIEW

A PREPRINT

Salim Dridi
contact.salimdridi@gmail.com - salimdridi.info

December 28, 2021

ABSTRACT

Supervised Learning, Unsupervised Learning, and Reinforcement Learning (RL) are the three main categories of Machine learning (ML). Supervised learning involves pre-training a model on a labeled dataset. On the other hand, in unsupervised learning, the model is trained on unlabeled data.

In contrast, RL is motivated by evaluating feedback rather than labels. Here, the agent learns the optimal path for solving sequential decision-making problems by interacting with the environment and taking the most suitable course of action in a given situation in order to maximize the reward. The RL agent makes its own decisions on how to complete a task. Additionally, there is no training data; as a result, the agent learns by accumulating experience. RL has applications in a wide variety of fields, including natural sciences, transportation, finance, and engineering, as well as smart grids, robotics, and healthcare.

Numerous RL approaches and algorithms for tackling real-world problems have been introduced over the previous decade. Among the well-known and extensively used methods are: the Markov Decision Process, Dynamic Programming, Temporal Difference, Monte Carlo, and Q-learning. The purpose of this survey is to examine RL and conduct a comprehensive review of the literature, methodologies and algorithms, performance metrics used to evaluate the RL agent, and the merits and demerits of numerous studies that used RL. This paper will point researchers in new directions and enable them to compare the efficacy and effectiveness of widely used RL algorithms.

Keywords Reinforcement Learning · Literature Review · Survey · Machine Learning · Reinforcement Learning approaches

Table of Contents

List of Figures

List of Tables

1 Introduction

ML, due to the concepts it inherits, can be regarded a subfield of Artificial Intelligence (AI). It enables prediction; for this purpose, its basic building blocks are algorithms. ML enables systems to learn on their own rather than being explicitly programmed to do so, resulting in more intelligent behavior. It generates data-driven predictions by developing models that discover patterns in historical data and utilize those patterns to generate predictions [1] [2]. The general architecture of ML is depicted in figure 1 and consists of several steps: business understanding (understanding and knowledge of the domain), data acquisition and understanding (gathering and understanding data), modeling (which entails feature engineering, model training, and evaluation), and deployment (deploy the model on the cloud).

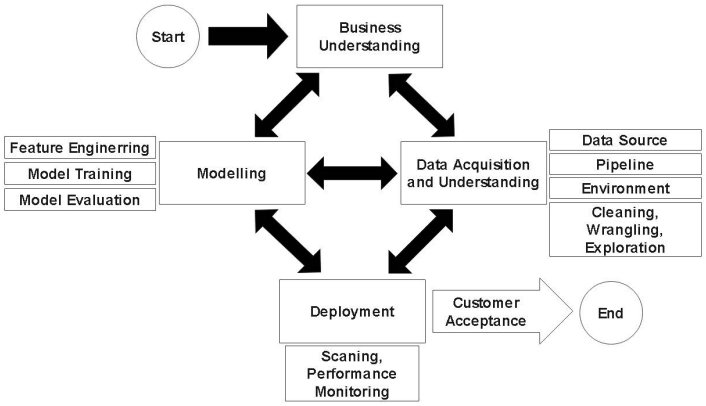

Figure 1: Basic Architecture of ML

Supervised Learning, Unsupervised Learning, and RL are the three broad categories of ML. In Supervised Learning, the model is trained on labeled data and is then used to generate predictions on unlabeled data [3]. In unsupervised learning, a model is trained on unlabeled data and said model automatically learns from that data by extracting features and patterns. In RL, an agent is trained on the environment and this enables said agent to find the optimum solution and accomplish a goal in a complex situation [4].

1.1 Reinforcement Learning (RL)

RL helps agents interact with their environment in an efficient manner, enabling consecutive decisions to be made. It promotes behavioral decision-making through the use of interaction experience and the subsequent evaluation of feedback [5]. The agent chooses an action based on the current state and receives evaluative feedback or reward and a new state. Then, it attempts to learn an optimal path in order to maximize the reward it receives over time by acquiring knowledge from its experiences rather than through any instructions. RL is distinct from classical supervised learning, which makes extensive use of labels and exhaustive reward signals. In contrast, RL relies on optimized sequential decisions [6]. Figure 2 illustrates the fundamental structure of RL.

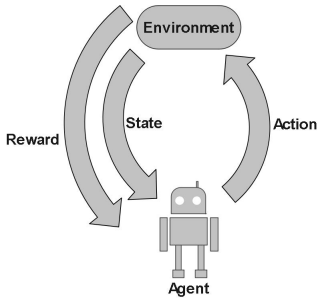

Figure 2: Basic Structure of RL

The terms used in RL are: environment, agent, reward, state, action, policy, value, and value function. The description of each of these terms is presented in the following table.

Term	Description
Agent	It is an entity that performs a specific action in an environment in order to receive a reward
Environment	It is a scenario or a situation that an agent has to face
State	State is referred as the current situation in which the agent is present
Action	Action is a specific task or move performed by an agent in an environment
Reward	Reward is a return that is given to an agent on performing some specific task or action in an environment
Policy	It is a strategy or a path chosen by the agent based on current state in order to decide the next task or action
Value	Value is a long-term reward, or a reward with a discount
Value function	It is the total reward and identifies the value of a state

Table 1: Basic Terms Used in a RL Model

The basic architecture of RL is presented in figure 3.

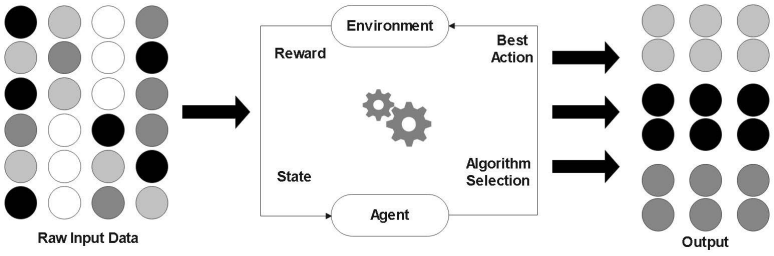

Figure 3: Basic Architecture of RL

RL is subdivided into two categories: Positive RL and Negative RL. Positive RL involves adding a reinforcing stimulus following a behavior to make it more likely that the behavior will reoccur. Positive RL maximizes performance and sustains variation for a more extended period. However, it may cause over-optimization. In contrast, negative RL involves the removal of something negative to strengthen a desired behavior. In other words, it strengthens an agent's behavior based on some adverse event that should have been avoided or stopped and helps to identify the minimum performance criteria.

1.1.1 Learning Models in RL

Essentially, there are two learning models in RL: Markov Decision Process and Q learning.

1.1.1.1 Markov Decision Process (MDP) MDP is a widely used probabilistic RL model for sequential decision problems and is a stochastic control process. In this model, the preceding states depend on the current state and action. MDP can be implemented using either dynamic or linear programming [7]. MDP is represented as a set of:

$$MDP = (S, A, r, T, \gamma)$$
where S represents a set of states;
A represents the set of actions;
$r = SxA \rightarrow R$ is a reward for taking an action in a state;
$T = SxAxS \rightarrow R$ a state transition function;
γ = a discount factor which indicates that a reward received in the future is less worthy than immediate reward

In MDP, the problem is solved by searching for an optimal policy that maps each state to an optimal action in accordance with a discounted long-term expected reward.

1.1.1.2 Q-Learning Q-learning is another important and widely adopted learning model. It assesses the quality of the action taken, hence named Q (quality) learning. It is also known as off-policy RL because it learns the action that is outside the current policy. In Q learning, a Q table has is created to store [state, action] values. This Q table is a reference one for the agent to choose the best action based on the Q-values. Then, the agent interacts with the environment and updates the Q-values or state-action pairs in the Q-table. The actions can be taken as exploiting or exploring. When an agent uses Q-table to view all Q-values and chooses the action based on the maximum q-value, it is known as exploiting. In contrast, when the agent takes random actions, it is known as exploring. Exploiting depends on maximum future reward, whereas exploring depends on acting randomly [8].

The following section presents the literature review of the topic, Section 3 provides the proposed methodology for conducting this systematic literature review (SLR), Section 4 provides the results of the study and the related discussion, and Section 5 presents the conclusion.

2 Literature Review

Numerous studies on RL have been written and implemented by the research community and practitioners. Additionally, there are already various surveys on RL. For instance, study [9] gives a survey on the use of RL techniques in healthcare. They categorize a number of healthcare studies into: dynamic treatment regimes, automated medical diagnosis, and other general domains. However, their work is limited to the healthcare domain. The authors of Paper [10] conducted a survey on RL algorithms where they described and demonstrated the operation of: Markov Decision Process, Monte Carlo methods, Q-learning, Temporal Difference, and Stochastic Approximation. They categorized RL algorithms into three classes: Markov decision processes, value prediction problems, and control. Their survey includes an overview of each aforementioned algorithm.

Study [11] provides a brief survey of Deep RL (DRL) and discusses the fundamentals of RL, policy and value-based methods, and essential algorithms in DRL such as: TRPO (Trust region policy optimization), DQN (Deep Q networks), and Asynchronous Critic. Furthermore, this study emphasized the merits of neural networks in RL. Study [12] is a survey of the applications of RL. The authors discuss a variety of research projects in recommender systems, computer systems, energy, finance, transportation, healthcare, and robotics. In their survey, they discuss several real-time applications of RL, including Horizon—an open-source RL platform—smart grids, data center cooling, medical picture report production, ridesharing, and order dispatching. Finally, the authors of research [13] conducted a survey on RL that was influenced by natural language processing (NLP) by learning from textual domain knowledge, text games, and following instructions. According to this study, NLP techniques can be utilized for RL tasks.

3 Methodology of this Systematic Literature Review

A SLR is a type of study that tries to identify and analyze existing literature on a certain subject. It is conducted formally and methodically, and is objective and reproducible. We conducted this SLR using the principles established by Barbara Ann Kitchenham [14].

This SLR will follow Kitchenham's guidelines and well-defined steps. The first stage is to define the research questions and then analyze the collected data to answer them. The second phase entails defining the search procedure. This study has included conference proceedings and journal articles dating back to 2011. "IEEE Xplore," "the Association for Computing Machinery (ACM)," and "Science Direct Elsevier" were the three databases used to look for papers. In the third phase, we defined the search strings that were used to retrieve related works from these databases. Then, the papers were subjected to inclusion and exclusion criteria: we included papers that were relevant to

the topic and have a publication date no older than 2011 and excluded all others. Finally comes Data Extraction which involves extracting data from the selected list of articles that address the three research questions.

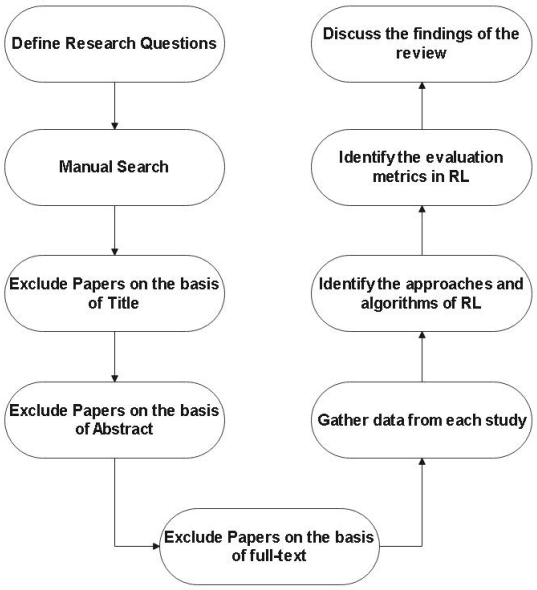

Figure 4: Methodology Used for the SLR on RL

3.1 Research Questions

The following are the Research Questions (RQ) formulated for this study.

RQ1: What are the approaches/algorithms that are used for problem-solving in RL?

RQ2: What are the widely used evaluation metrics for measuring the performance of the employed RL model?

RQ3: What are the merits and demerits of each study in RL?

3.2 Search Strategy

Once the key terms were identified, we initiated the search process. From these different terms, we formulated the query strings.

3.3 Query Strings

To formulate query strings, we followed the three databases' (ACM Digital Library, IEEE Xplore, and Science Direct Elsevier) guidelines in using Boolean operators, synonyms, and different terms. For our initial search strings, thousands of papers appeared, so we had to refine them and use the Advanced Search option so a lower number of papers would be returned to us. Table 2 below shows the final Search strings that were used.

Database Name	Search String
IEEE Xplore	((reinforcement learning) AND (reinforcement machine learning approaches))
ACM Digital Library	acmdlTitle:(reinforcement learning) AND (+"algorithms and approaches")
Science Direct	reinforcement learning algorithms AND reinforcement learning approaches

Table 2: Search Strings

3.4 Search Results

To retrieve the articles, we ran the queries in August 2021. The result of the query strings is shown in Table 3. Then, for the citations and bibliographies, we imported these papers into Mendeley Library. After eliminating duplicates, we ended up with a total of 136 articles.

Database Name	Search String	Years	Number of Papers
IEEE Xplore	((reinforcement learning) AND (reinforcement machine learning approaches))	2011-2021	46
ACM Digital Library	acmdlTitle:(reinforcement learning) AND (+"algorithms and approaches")	2011-2021	36
Science Direct	reinforcement learning algorithms AND reinforcement learning approaches	2011-2021	55

Table 3: Search Results

3.5 Study Selection Criteria

Based on our stated research questions and SLR topic, inclusion and exclusion criteria were defined as follows.

Inclusion Criteria

Papers were included based on the following assumptions:

1. Those which were published between 1st January 2011 and 1st August 2021.
2. Those which contain a RL Approach.
3. Those that discuss RL Algorithms.

Exclusion Criteria

Papers were excluded based on the following assumptions:

1. Those which were published before 2011.
2. Those which do not focus on RL.
3. Those which are describing RL but did not focus on any approach or algorithm of RL.

3.6 Search Process

We used the following keywords to conduct our search in the aforementioned databases: "reinforcement machine learning", "reinforcement learning ", "reinforcement learning algorithm", "reinforcement learning approach." Then, we set the year filter to exclude works published prior to 2011. Following that, we assessed the papers that were relevant to our investigation. Several were eliminated solely on the Title, while others were eliminated after reading the Abstract. Finally, as shown in Table 4, we ended up with 46 papers for our survey.

Database Name	Years	No. of Papers	No. of Papers Excluded on Basis of Title	No. of Papers Excluded on Basis of Abstract	No. of Papers Selected
IEEE Xplore	2011-2021	46	16	12	18
ACM Digital Library	2011-2021	36	10	11	15
Science Direct	2011-2021	55	20	22	13

Table 4: Number of Papers Selected After Applying Inclusion and Exclusion Criteria

Figure 5 summarizes the strategy used to conduct this literature review on RL. As noted previously, three databases were used to choose the publications. Following that, we obtained a total of 136 articles from three databases. Then, as discussed, we applied the inclusion and exclusion criteria. After applying the inclusion and exclusion criteria, 46 papers remained that also matched our ten-year time window (1 January 2011 - 1 August 2021). The next section contains the results of the study.

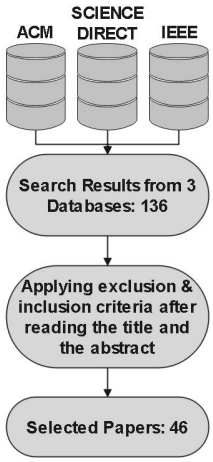

Figure 5: Scope of the Study

4 Results and Discussion

In this section, the results corresponding to each formulated research question are provided.

RQ1: What are the approaches/algorithms that are used for problem-solving in RL?

Many RL approaches and algorithms have been proposed since the last decade and we will discuss many of them in this section. From all the studies we looked at, we found that different RL algorithms vary from employing value-based to policy-based methods. Figure 6 presents a list of RL algorithms and approaches.

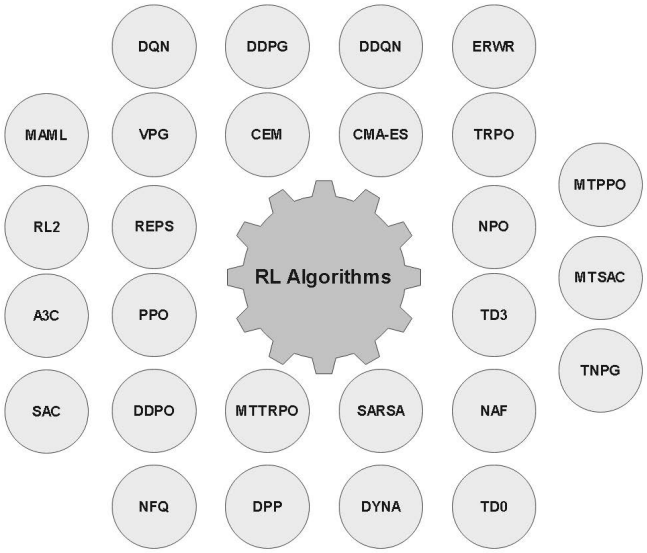

Figure 6: List of RL Algorithms

4.1 Cross-Entropy Method (CEM)

Cross entropy method (CEM) is a straightforward EDA(Estimation of Distribution Algorithm)-based method in which a certain number of specific high-performing agents are fixed in order to reach a certain value, K_e. The working of this algorithm is such that after analyzing each selected agent, the fittest ones are shortlisted to compute the population's new mean and variance. With the help of these results, the next generation is sampled along with variance e, to avoid premature convergence. [15][16][17]

4.2 Covariance Matrix Adaptation Evaluation Strategy (CMA-ES)

As with the preceding CEM algorithm, CMA-ES is also a RL algorithm that selects a subset of the best performing agents in order to find a certain value of k_e. Here too, the new generation's covariance and mean are derived from these agents. However, in comparison to CEM, the workings of CMA-ES is more straight forward. The chosen agents are ranked according to their performance, and weights are applied accordingly. The results are influenced by these weights. Additionally, measures referred to as 'evolutionary paths' are utilized to accumulate search direction for the next generations. However, one of the minor variations between CEM and CMA-ES algorithms is how the covariance matrix is updated. In the standard formulation method, CEM employs a new estimate of the mean μ to calculate the new $\sum$. In contrast, CMA-ES makes use of the current μ (the one used to sample the current generation). The formulation of the new $\sum$ in CMA-ES is based on the formula given below:

$$\sum_{new} = \sum_{i=1}^{K_e} \lambda_i (z_i - \mu_{old})^2 + I$$

Where the square of the vectors denotes the vectors of the square of the coordinates.

4.3 Deep Deterministic Policy Gradient (DDPG) and Twin Delayed DDPG (TD3)

DDPG and TD3 are two Deep-RL algorithms that are actor-critic, off-policy, and sample efficient. DDPG is a hybrid of DQN and QAC that features deterministic policy and off-policy update using a replay buffer. It use deterministic policy as an approximate Q-value maximizer over action space. In exploration, it employs Gaussian noise for stochastic actions, and uses target networks and a delayed update [46]. On the other hand, TD3 is a more advanced algorithm based on DDPG with twin action-value networks and delayed updating for the policy and the target networks. TD3 is based on Double Q-Learning and uses smoothing regularization for the target polict [46]. DDPG features a few flaws and instabilities, which are due in part to an overestimation bias in critic updates. It is known to be difficult to tune due

to its sensitivity to hyper-parameter settings. The availability of efficiently-tuned code baselines that incorporate several advanced mechanisms can help address these issues. In contrast, the TD3 algorithm resolves the issues with DDPG by employing two critics and taking the lowest estimate of the action values into account in the update mechanisms [15]. The figure below illustrates how both algorithms work:

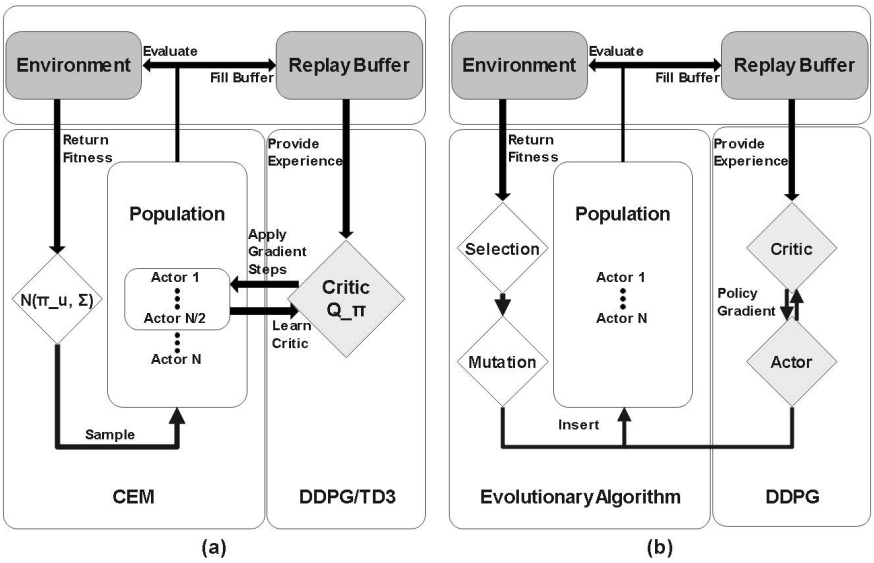

Figure 7: Workings of DDPG and TD3

4.4 Deep Q Network (DQN) and Double Deep Q Network (DDQN)

DQN is a RL algorithm that combines Q-Learning and DNNs in order to enable reinforcement learning to work in high-dimensional and complicated contexts such as video gaming and robotics. However, DQN has certain shortcomings, which DDQN fixes. It compensates for the DQN algorithm's occasional tendency to overestimate the values associated with specific actions are utilized to approximate the state-action value function. Thus, the DQN can be trained if the prediction error is kept to a minimum. Although the Deep Q Learning algorithm is effective, it is known to have significant flaws, including an overestimation of action values under certain scenarios. As a result, an advanced algorithm, dubbed Double Deep Q-learning, was built by researchers to solve these problems. Since the max operator in both Q-learning (DQL) and DDQN (3) selects and evaluates an action using the same values, it is possible to pick overestimated values, resulting in overly optimistic value estimates. Thus, the idea of Double Deep Q-learning is to minimize overestimation by decomposing the target's maximum operation into action selection and action evaluation. In comparison to DQN, Double DQN differs only in the phase of updating the Q-value [18][19].

4.5 Markov Decision Processes (MDP)

MDP is a deep learning algorithm that is sometimes referred to as a Triplet. This process is defined by a finite set of actions and states. Each time the agent observes a state and takes an action, intermediary costs are initiated and must be minimized. The cost and successor state are determined by the current state as well as the chosen action. Successor generations may have certain probabilities that are based on the level of uncertainty we have about the environment in which the search occurs. For example, an action may occasionally fail to reach the desired target state, preferring instead to remain in the present state with a small probability. A MDP problem is a tuple; S denotes the underlying state space, A denotes a set of actions, r is the cost or immediate reward function, and T is the probability that action a in state u will result in state v. In some MDPs, additional costs c are incurred when a goal state is reached. The objective is to reduce the anticipated accumulated costs or, alternatively, to maximize the expected accumulated rewards [19]. POMDPs and SMDPs have also emerged from MDPs with similar functionalities along with slight variations.

4.6 Feature Learning/Representation Learning

Feature learning, also sometimes termed Representation Learning, refers to a group of techniques that enable systems to automatically identify the mechanisms required to process raw data. Generally, machines are unable to see, hear, or

understand directly. As a result, we must construct a data representation understandable by machines, which is the fundamental goal of the feature learning approach. This is critical because it improves machines' ability to process large numbers of features and makes deep learning more powerful when dealing with unstructured data. However, the disadvantage of such a complicated method is that it may be overkill for simpler problems that require access to a huge amount of data to be effective. While ML makes use of relatively simple concepts, feature learning makes use of ANNs, which try to mimic the way humans think and learn. It can be used to tackle pattern recognition problems without requiring human intervention. Deep learning is powered by ANNs with numerous layers.

4.7 State-Action-Reward-State-Action (SARSA) and SARSA(λ)

SARSA is another RL Algorithm that has proven to be simple and easy to implement. Despite its numerous advantages, relative to other algorithms, it has a drawback related to producing less efficient scaling results. SARSA(λ), on the other hand, exhibits a small amount of performance fluctuation. The latter uses eligibility traces which means that past state action pairs still contribute to the current state, so even if the learning rate α is reset immediately after the change in order to allow a faster convergence, it seems not enough.

4.8 DYNA, DYNA Q, and DYNA Q+

Dyna Algorithms have several limitations and seem to perform much below average while working on deterministic problems. In the presence of changes, none of them seems to perform effectively. These algorithms use a model of the environment to update Q(x, a) or P(x, a), meaning that after each interaction with the environment, they perform periodic iterations using simulated triplets (x, a, r). In the presence of changes, this results in obsolete information being reused several times until sufficient fundamental interactions with the environment occur, and the model is also updated. This is the main reason why each Dyna algorithm needs more iterations after the change than its corresponding learning algorithm. Dyna Q Learning only updates a single entry of Q(x, a) at each simulated iteration, which could explain why noise does not corrupt Q(x, a) too much and why this algorithm performs well in the presence of uncertainty. Noise, in this case, is added at a single entry of Q(x, a), rather than to the whole matrix, at each iteration [20].

4.9 Temporal Difference (TD)

Temporal difference (TD) learning is an RL approach for learning about the prediction techniques that depend on the future values of a provided signal. The prediction at any given time step is updated to bring it closer to the prediction of the same quantity at the next time step. Each sample of the TD Learning technique consists of a few steps and not on the whole trajectory.

4.10 Behavioral Cloning

This is an approach by which human sub-cognitive expertise can be identified, recorded, and then reproduced with the help of a computer program. This process of reconstructing a skill from the behavioral traces of a human object uses the means of Machine Learning techniques. As human subjects perform any skill, their actions are recorded along with the situation that raised that specific action. The data of these skill records are used as input to a learning program. We can say that this process works by cloning the behavior of the object (human).

4.11 Monte Carlo (MC)

The traditional Markov chain is apart from the numerous advantages of Bayesian inference, such as quantification of uncertainty, accurate average prediction, and preventing over-fitting. The Monte Carlo (MCMC) approach has been said to be rather unsuitable for large-scale problems because it requires processing the entire dataset per iteration rather than only using a small random mini-batch as performed in the stochastic gradient optimization. The first attempt towards the scalable MC method based on stochastic gradients is the stochastic gradient Langevin dynamics (SGLD) proposed by The and Welling in 2011. This originated from the Langevin Monte Carlo method; SGLD achieved O(n) computation per iteration using stochastic gradients estimated using mini-batches and skipping the Metropolis-Hastings accept the rejected test. The approach has certain limitations, such as the poor mixing rate and the arbitrarily large bias that occurred while using the large step sizes.

4.12 Q-Learning and Q-Learning (λ)

Q-learning is a RL algorithm used to learn the value of an action in a particular state. It does not need a model of the environment as input and hence, therefore, called model-free. It can handle problems with stochastic transitions and

rewards without the need for adaptations. Q-learning uses an optimal policy for any finite Markov decision process (FMDP) that maximizes the expected value of total reward over any or all successive steps, starting from any current state. This algorithm is helpful and can identify an optimal action-selection policy for any given FMDP, given infinite exploration time and a partly random policy. "Q" refers to the function computed by the algorithm—expected rewards for an action taken in a given state.

4.13 Episodic Reward Weighted Regression (ERWR)

ERWR stands for Episodic Reward Weighted Regression Algorithm. This is a DRL Algorithm. This is an extension of the original RWR algorithm, which is based on a linear policy to solve the immediate rewards learning problem. The extension implemented in this algorithm applies RWR to episodic RL. ERWR is an RL algorithm and is known as the only gradient-based algorithm. The implementation of this algorithm does not require any hyper-parameter tuning. This is capable of solving some basic tasks providing us with efficient results. The drawback of this algorithm is that it fails to solve complex and more challenging tasks, including locomotion. However, it is observed empirically that RWR shows faster initial improvement, which is followed by a noticeable slow-down [22].

4.14 Natural Policy Gradient (NPO) - Optimization

This method comprises a natural gradient method that shows the steepest descent direction based on the underlying structure of the parameter space. Although, gradient methods are not suitable for significant changes for the parametric values. However, studies show that the natural gradient is choosing a greedy optimal action rather than looking for algorithms that provide effective results. These greedy optimal actions would be chosen under one improvement step of policy iteration with approximate, compatible value functions [21].

4.15 Proximal Policy Optimization (PPO)

PPO in an RL Algorithm that is an alternative between data sampling through environmental interactions and optimizing a "surrogate" objective function using a stochastic gradient ascent. In contrast, the standard policy gradient methods perform one gradient update per data sample. This algorithm has a straightforward implementation and can be applied to multiple scenarios due to the high level of generality. Comparative to other algorithms, the PPO has a better sample complexity.

4.16 Actor-Critic Agent (A3C)

This is conceptually a straightforward and lightweight framework used for deep RL. This method uses an asynchronous gradient descent for the optimization of deep neural network controllers. A research was conducted based on asynchronous variants of four standard RL algorithms. The results show that parallel actor-learners have a stabilization impact on training and allow all four methods to train the neural network controllers successfully. Results show that the best performing method, an asynchronous variant of actor-critic, outperforms the current best-performing algorithms. Furthermore, studies have shown that asynchronous actor-critic succeeds on a wide variety of continuous motor control problems.

4.17 Soft Actor-Critic (SAC) and Multi-Task SAC

This is a model-free deep RL algorithm based on DDPG using an additional soft entropy term for boosting exploration. These algorithms can be demonstrated on a wide range of challenging decision control and decision-making tasks. However, such methods usually suffer from two challenges: one is a very high sample complexity and complex properties of convergence, which gives rise to hyper-parameter tuning problems. These limitations cannot be ignored because they severely limit the applications of such methods in complex, real-world domains. Prior deep RL methods based on this framework have been formulated as Q-learning methods.

4.18 Model-Agnostic Meta-Learning (MAML)

This algorithm for meta-learning is based on model-agnostics so that it can be made compatible with any model trained with gradient descent. Therefore it applies to a wide range of different RL problems, including regression, classification, and RL. The main aim of meta-learning is to simulate a model on a variety of learning tasks. It can solve new learning tasks using only a small amount of training samples. Research suggests that this approach leads to state-of-the-art performance and is therefore widely applicable, having efficient results.

4.19 Normalized Advancement Functions (NAF)

NAF is also a model-free RL technique and has been successfully applied to numerous challenging RL problems and has recently been extended to handle large-value functions and neural network policies. However, the sample complexity of model-free algorithms, especially while using high-dimensional function approximators, tends to limit their applications to physical systems. Normalized Advancement Functions (NAF) representation also allows us to apply Q-learning and experience replay to continuous tasks and thus improve performance on a set of simulated robotic control tasks. Hence, the iteratively refitted local linear models are especially effective for these scenarios and demonstrate comparatively faster learning on domains where such models can be applied.

4.20 Task Embedding

This is one of the methods used to make the applicability of neural networks adaptable to emerging scenarios and new tasks. This also is a promising method for the reduction of limitations that are faced by other RL algorithms. However, the network is usually programmed and trained to perform the required functions for any specific task. This method is used to provide vectorial representations as well, as we know that task. Allocation is one of the significant problems for fundamental combinatorial optimization. However, the proposed algorithm has higher applicability and is helpful in such kinds of scenarios as well.

4.21 Meta-RL

Meta-learning is usually a branch of metacognition that is concerned with learning with the help of one's learning and learning processes. This term comes from the meta-prefix's modern meaning of an abstract recursion. However, Meta RL or meta-RL is also an RL algorithm. This algorithm is helpful to provide experience from previous learning tasks to learn how to learn new tasks quickly. We can say that this approach provides learning about how to learn. This also provides process automation during task design by providing a meta-learning algorithm that does not need manual meta-training tasks.

4.22 Relative Entropy Policy Search (REPS)

This RL algorithm has an advantage over other RL algorithms and approaches in that it helps to limit the information loss as per every iteration cycle. The main aim of this algorithm that leverages the learning process and also ensures a smooth learning process. It is observed that REPS is primarily liable to early convergence to local optima when dealing with continuous states and actions. Its outcome is greatly affected by the performance of the initial policy. This, indeed, leads to a bad performance on average. However, under particular initial settings, the algorithm can perform much better than others.

4.23 Truncated Natural Policy Gradient (TNPG)

In the domain of neural networks and policies with thousands and millions of parameters or more, generic Natural Policy Gradient usually requires the unavoidable cost of computation by forming and inverting the empirical FIM. Instead, we have the Truncated Natural Policy Gradient (TNPG) approach that will help us overcome these drawbacks. Therefore, this process is preferable and makes it easy and practical to apply natural gradients in policy search settings even along with high-dimensional parameters. Researchers say that both the TNPG and TRPO algorithms can outperform other algorithms of the same kind by a large margin in most of the tasks and confirm that constraining the change in the policy distribution results in more stable learning.

4.24 Trust Region Policy Optimization (TRPO) and Multi-Task TRPO

This algorithm has the benefit that we have more precise control on the expected policy improvement compared to TNPG with the help of the introduction of a parameter known as surrogate loss. At every single iteration, the algorithm is capable of solving the following constrained optimization problem [23]:

$$maximize_{\theta} \rightarrow \mathbb{E}_{s\sim\rho\theta_k}, a \sim \pi_{\theta_k}\left[\frac{\pi_{\theta}(a|s)}{\pi_{\theta_k}(a|s)}A_{\theta_k}(s,a)\right]$$

$$s.t. \rightarrow E_{s\sim\rho\theta_k}\left[D_{KL}(\pi_{\theta_k}(\cdot|s)||\pi_{\theta}(\cdot|s))\right] \leq \delta_{KL}$$

Where $\rho\theta = \rho\pi_\theta$ is the discounted state-visitation frequencies. This is also known as the advantage function. This is estimated by empirical return minus baseline.

RQ2: What are the widely used evaluation metrics for measuring the performance of the employed RL model?

According to our survey, several evaluation metrics are used by researchers for the performance evaluation of several RL models. table 5 presents a list of widely used evaluation metrics.

No	Evaluation Metric	Description
1	Dispersion across Time (DT)	Dispersion across Time (DT) is measured by isolating higher-frequency variability instead of relying on longer-term trends.To avoid the metrics from getting influenced by a positive trend detrending is applied. Detrending is a statistical technique that involves removing the effects of accumulating data sets from a trend to show only the absolute changes in values and to identify potential repeating patterns. The final measure of DT consists of an interquartile range (IQR) within a sliding window along the detrended training curve.
2	Short-term Risk across Time (SRT)	This metric gives the worst-case expected drop in performance during training, from one point of evaluation to the next. To do this, CVaR is applied to the changes in performance from one evaluation point to the next. SRT is calculated as follows: **1)** Compute the differences between two-time points on each training run **2)** Normalise the differences by the distance between time-points to ensure invariance to evaluation frequency **3)** Obtain the distribution of these differences and find the α-quantile **4)** Compute the expected value of the distribution below the α-quantile
3	Long-term Risk across Time (LRT)	This metric helps in monitoring the performance relative to the highest peak so far and can be used to capture unusually large drops that occur over longer timescales (drawdown). For this measure, CVaR is applied to the drawdown time.
4	Dispersion across Runs (DR)	It is measured by taking the variance or standard deviation across training runs at a set of evaluation points. First, low-pass filtering is performed on the training data to filter out high-frequency variability within runs. The variance or standard deviation is replaced with IQR.
5	Risk across Runs (RR)	CVaR is applied to the final performance of all the training runs. Using this metric gives an idea of the performance of the worst runs.
6	Dispersion across Fixed-Policy Rollouts (DF)	To compute this metric, the IQR is calculated on the performance of the rollouts. This helps in evaluating a fixed policy for checking the variability in performance when the same policy is rolled out multiple times.

Continued on next page

Table 5 Continued from previous page

No	Evaluation Metric	Description
7	Risk across Fixed-Policy Rollouts (RF)	This metric is similar to Dispersion across Fixed-Policy Rollouts except that CVaR is applied on the rollout performances.

Table 5: Widely Used Performance Metrics in RL

RQ3: What are the merits and demerits of each study in RL?

Table 6 presents several studies, the approach employed and the merits and demerits associated with each of the study.

No	Study Title	Approach	Merit	Demerit	Ref
1	IntelliLight: A Reinforcement Learning Approach for Intelligent Traffic Light Control	Q learning	Successfully solve the problem of traffic light control, which are helpful in generating traffic rules	It is limited to only two-phase traffic light only, no filed study and the feedback is simulated	[24]
2	Resource Management with Deep Reinforcement Learning	Deep RM	Deep RM translates the packing tasks with multiple demands into learning problem and adapts several conditions and learn sensible strategies	DeepRM uses a small or finite time horizon however the underlying optimization problem has an infinite time horizon	[25]
3	Energy-Efficient UAV Control for Effective and Fair Communication Coverage: A Deep Reinforcement Learning Approach	DRL	A novel method that maximizes energy efficiency and outperforms other models in terms of fairness, coverage and energy consumption	Scalability issue	[26]
4	Smart Resource Allocation for Mobile Edge Computing: A Deep Reinforcement Learning Approach	DRL	Allocates computing resources efficiently and minimize the service time in a varying MEC environment	Average service time is high	[27]

Continued on next page

Table 6 Continued from previous page

No	Study Title	Approach	Merit	Demerit	Ref
5	Integrated Networking, Caching, and Computing for Connected Vehicles: A Deep Reinforcement Learning Approach	DQN	A novel DRL framework for efficient resource allocation, caching and dynamic orchestration of networking	High complexity	[28]
6	Practical Deep Reinforcement Learning Approach for Stock Trading	DDPG	The proposed model results in optimized stock trading and maximizing investment return. It also outperformed the other models in terms of cumulative returns and share ratio.	Small scale data is used, lack prediction schemes	[29]
7	Traffic and Computation Co-Offloading With Reinforcement Learning in Fog Computing for Industrial Applications	MDP	The proposed framework efficiently solves the offloading decision problem and outperforms other state of the art models in mobile edge networking	Interoperability of edge devices is not considered	[30]
8	ontrol of microbial co-cultures in bioreactors	Q learning	Successfully controlled the population of microbial communities to the target level	High computational time	[31]
9	Reinforcement Learning for Dynamic Microfluidic Control	DQN MFEC	Proposed framework successfully controls and processes each experiment in microfluidic experimentation and yields high throughput	High processing time	[32]
10	Deep reinforcement learning for De-Novo drug design	DRL	Designed chemical libraries with some specific properties. The model is capable of generating chemical libraries for compounds with specific properties	High complexity	[33]

Continued on next page

Table 6 Continued from previous page

No	Study Title	Approach	Merit	Demerit	Ref
11	A Text-based Deep Reinforcement Learning Framework for Interactive Recommendation	DDPG NPO	Efficiently address the issues related to interactive recommendation systems by removing data sparsity in time efficient manner	High sample complexity can reduce the performance	[34]
12	Automated Deep Reinforcement Learning Environment for Hardware of a Modular Legged Robot	DRL TRPO DDPG	Effectively learn locomotion skill of highly stochastic hardware of a modular legged robot	Limited motions and limited rewarding terms	[35]
13	Stock Market Trading Agent Using On-policy Reinforcement Learning Algorithms	MDP VPG TRPO PPO	The proposed framework maximizes the profits of stock market trading	The reward function can be further optimized by including risk factor	[36]
14	Mobile Robot Path Planning Based on Improved DDPG Reinforcement Learning Algorithm	DDPG	Propose a robot path planning method with less error and trials, fast convergence and less computation time	High sample complexity	[37]
15	Dynamic Pricing Strategy of Electric Vehicle Aggregators Based on DDPG Reinforcement Learning Algorithm	DDPG	Minimize the load fluctuation of power grid and guide the charging behavior more effectively	High computational cost	[38]
16	Motion Control for Biped Robot via DDPG-based Deep Reinforcement Learning	DDPG	The used methodology controls the fall overs of a robot and helps in a stable walk	Sparse rewards	[39]
17	Resource Pricing and Allocation in MEC Enabled Blockchain Systems: An A3C Deep Reinforcement Learning Approach	A3C	A3C provides a good balance between reward and risk and outperforms the baseline algorithms	High complexity	[40]

Continued on next page

Table 6 Continued from previous page

No	Study Title	Approach	Merit	Demerit	Ref
18	A composite learning method for multi-ship collision avoidance based on reinforcement learning and inverse control	A3C	Successful simulations for ship collision avoidance	High processing time	[41]
19	Improving Search Through A3C Reinforcement Learning Based Conversational Agent	A3C	Proposed a stochastic virtual user by providing a labeled conversational data for optimized searching	Sample complexity	[42]
20	Design and implementation of an adaptive cruise control system based on supervised actor-critic learning	SAC	Adaptive cruise control system with high performance considering the emergency braking scenario and road conditions.	Poor stability during training	[43]
21	Task Offloading and Resource Allocation for Mobile Edge Computing by Deep Reinforcement Learning Based on SARSA	SARSA	Successfully resolve the resource allocation problem with optimal offloading decision, minimizing the system cost	The satisfactory results were based on specific value of €	[44]
22	Dyna-H: A heuristic planning reinforcement learning algorithm applied to role-playing game strategy decision systems	DYNA-Q	Proposed a DYNA agent that finds an optimal path in role-playing game and yield superior results	Uniformly samples the user goals	[45]

Table 6: Approaches, Merits, and Demerits of RL Studies

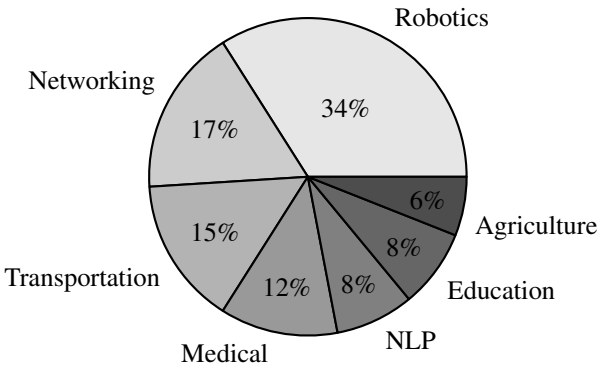

Figure 8: Taxonomy of Papers Based on Research Areas

5 Conclusion

RL is one of the broad classes of machine learning. It enables behavioural decision-making using interaction experience and then evaluating feedback. There is no need for a supervisor; the agent learns based on their experience and is rewarded for choosing a suitable action. The ultimate goal of RL is to find and learn an optimal path or a policy to perform some task and increase the reward over time. It can be categorized as positive RL or negative RL. Since the last decade, several RL approaches have been proposed. The application of RL ranges from Horizon- an open-source RL platform, smart grids, data center cooling, medical image report generation to ridesharing and order dispatching. This study is based on RL. This SLR follows a sequence of well-defined steps proposed by Kitchenham. This literature survey encompasses the RL approaches and algorithms and several evaluation metrics used for the performance evaluation of RL models. It also presents the merits and demerits of each study. This survey paper will help guide the researchers on which RL approach or algorithms to use for their problem-solving. Furthermore, which area of research needs more attention.

This study is limited to widely used RL algorithms only. It highlights the research papers of the past ten years and from three databases only. In future, we can involve more databases and more algorithms and approaches for better guidance.

References

[1] James Cussens, "Machine Learning," *IEEE Journal of Computing and Control*, Vol.7, No.4, pp.164-168, 1996.

[2] Muhammad, I., & Yan, Z., "Supervised Machine Learning Approaches A Survey," *ICTACT Journal on Soft Computing*, Vol.5, No.3, 2015.

[3] S. B. Kotsiantis, "Supervised Machine Learning: A Review of Classification Techniques," *Informatica*, Vol.31, No.3, pp.249-268, 2007.

[4] Richard S. Sutton and Andrew G. Barto, "Reinforcement Learning: An Introduction," *Cambridge, MA: MIT Press*, 1998.

[5] Wiering, M. A., & Van Otterlo, M., "Reinforcement learning," *Adaptation, learning, and optimization*, Vol.12, No.3, 2012.

[6] Kaelbling, L. P., Littman, M. L., & Moore, A. W.,"Reinforcement learning: A survey," *Journal of artificial intelligence research*, Vol.4, pp.237-285, 1996.

[7] Levin, E., Pieraccini, R., & Eckert, W., "Using Markov decision process for learning dialogue strategies," *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98*, Vol.1, pp.201-204, May 1998, Cat. No.98CH36181.

[8] Watkins, C. J., & Dayan, P., "Q-learning," *Machine learning*, Vol.8, No.3-4, pp.279-292, 1992.

[9] Yu, C., Liu, J., & Nemati, S., "Reinforcement learning in healthcare: A survey," *arXiv preprint*, arXiv:1908.08796, 2019.

[10] Szepesvári, C., "Algorithms for reinforcement learning," *Synthesis lectures on artificial intelligence and machine learning*, Vol.4, No.1, pp.1-103, 2010.

[11] Arulkumaran, K., Deisenroth, M. P., Brundage, M., & Bharath, A. A., "Deep reinforcement learning: A brief survey," *IEEE Signal Processing Magazine*, Vol.34, No.6, pp.26-38, 2017.

[12] Li, Y., "Reinforcement learning applications," *arXiv preprint*, arXiv:1908.06973, 2019.

[13] Luketina, J., Nardelli, N., Farquhar, G., Foerster, J., Andreas, J., Grefenstette, E., ... & Rocktäschel, T., "A survey of reinforcement learning informed by natural language," *arXiv preprint*, arXiv:1906.03926, 2019.

[14] B. Kitchenham, O. P. Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, "Systematic literature reviews in software engineering – A systematic literature review," *Inf. Softw. Technol.*, Vol.51, No.1, pp.7-15, 2009.

[15] Pourchot, A., & Sigaud, O., "CEM-RL: Combining evolutionary and gradient-based methods for policy search," *arXiv preprint*, arXiv:1810.01222, 2018.

[16] Liu, Z., Zhou, H., Chen, B., Zhong, S., Hebert, M., & Zhao, D., "Constrained Model-based Reinforcement Learning with Robust Cross-Entropy Method," *arXiv preprint*, arXiv:2010.07968, 2020.

[17] Joseph, A. G., & Bhatnagar, S., "An online prediction algorithm for reinforcement learning with linear function approximation using cross entropy method," *Machine Learning*, Vol.107, No.8, pp.1385-1429, 2018.

[18] Zhao, N., Liang, Y. C., Niyato, D., Pei, Y., & Jiang, Y., "Deep reinforcement learning for user association and resource allocation in heterogeneous networks," *2018 IEEE Global Communications Conference (GLOBECOM), IEEE*, pp.1-6, December 2018.

[19] Bester, C. J., James, S. D., & Konidaris, G. D., "Multi-pass q-networks for deep reinforcement learning with parameterised action spaces," *arXiv preprint*, arXiv:1905.04388, 2019.

[20] Casanueva, I., Budzianowski, P., Su, P. H., Mrkšić, N., Wen, T. H., Ultes, S., ... & Gašić, M., "A benchmarking environment for reinforcement learning based task oriented dialogue management," *arXiv preprint*, arXiv:1711.11023, 2017.

[21] Xu, Y., & Yoshimasa, T., "Parametrized control in soccer simulation with deep reinforcement learning," *The 22nd Game Programming Workshop 2017*, pp.208-214, November 2017.

[22] Wang, J., "Policy Hyperparameter Exploration for Behavioral Learning of Smartphone Robots," 2017.

[23] Duan, Y., Chen, X., Houthooft, R., Schulman, J. & Abbeel, P., "Benchmarking Deep Reinforcement Learning for Continuous Control," *Proceedings of The 33rd International Conference on Machine Learning, in Proceedings of Machine Learning Research*, Vol.48, pp.1329-1338, 2016.

[24] Wei, H., Zheng, G., Yao, H., & Li, Z., "Intellilight: A reinforcement learning approach for intelligent traffic light control," *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp.2496-2505, July 2018.

[25] Mao, H., Alizadeh, M., Menache, I., & Kandula, S., "Resource management with deep reinforcement learning," *Proceedings of the 15th ACM workshop on hot topics in networks*, pp.50-56, November 2016.

[26] Liu, C. H., Chen, Z., Tang, J., Xu, J., & Piao, C., "Energy-efficient UAV control for effective and fair communication coverage: A deep reinforcement learning approach," *IEEE Journal on Selected Areas in Communications*, Vol.36, No.9, pp.2059-2070, 2018.

[27] Wang, J., Zhao, L., Liu, J., & Kato, N., "Smart resource allocation for mobile edge computing: A deep reinforcement learning approach," *IEEE Transactions on emerging topics in computing*, 2019.

[28] He, Y., Zhao, N., & Yin, H., "Integrated networking, caching, and computing for connected vehicles: A deep reinforcement learning approach," *IEEE Transactions on Vehicular Technology*, Vol.67, No.1, pp.44-55, 2017.

[29] Xiong, Z., Liu, X. Y., Zhong, S., Yang, H., & Walid, A., "Practical deep reinforcement learning approach for stock trading," *arXiv preprint*, arXiv:1811.07522, 2018.

[30] Wang, Y., Wang, K., Huang, H., Miyazaki, T., & Guo, S., "Traffic and computation co-offloading with reinforcement learning in fog computing for industrial applications," *IEEE Transactions on Industrial Informatics*, Vol.15, No.2, pp.976-986, 2018.

[31] Treloar, N. J., Fedorec, A. J., Ingalls, B., & Barnes, C. P., "Deep reinforcement learning for the control of microbial co-cultures in bioreactors," *PLoS computational biology*, Vol.16, No.4, e1007783, 2020.

[32] Dressler, O. J., Howes, P. D., Choo, J., & deMello, A. J., "Reinforcement learning for dynamic microfluidic control," *ACS omega*, Vol.3, No.8, pp.10084-10091, 2018.

[33] Popova, M., Isayev, O., & Tropsha, A., "Deep reinforcement learning for de novo drug design," *Science advances*, Vol.4, No.7, eaap7885, 2018.

[34] Wang, C., Guo, Z., Li, J., Pan, P., & Li, G., U., "A Text-based Deep Reinforcement Learning Framework for Interactive Recommendation," *arXiv preprint*, arXiv:2004.06651, 2020.

[35] Ha, S., Kim, J., & Yamane, K., "Automated deep reinforcement learning environment for hardware of a modular legged robot," *2018 15th International Conference on Ubiquitous Robots (UR), IEEE*, pp.348-354, June 2018.

[36] Lele, S., Gangar, K., Daftary, H., & Dharkar, D., "Stock Market Trading Agent Using On-policy Reinforcement Learning Algorithms," *Available at SSRN 3582014*, 2020.

[37] Dong, Y., & Zou, X., "Mobile Robot Path Planning Based on Improved DDPG Reinforcement Learning Algorithm," *2020 IEEE 11th International Conference on Software Engineering and Service Science (ICSESS), IEEE*, pp.52-56, October 2020.

[38] Liu, D., Wang, W., Wang, L., Jia, H., & Shi, M., "Dynamic Pricing Strategy of Electric Vehicle Aggregators Based on DDPG Reinforcement Learning Algorithm," *IEEE Access*, Vol.9, pp.21556-21566, 2021.

[39] Wu, X., Liu, S., Zhang, T., Yang, L., Li, Y., & Wang, T., "Motion control for biped robot via DDPG-based deep reinforcement learning," *2018 WRC Symposium on Advanced Robotics and Automation (WRC SARA), IEEE*, pp.40-45, August 2018.

[40] Du, J., Cheng, W., Lu, G., Cao, H., Chu, X., Zhang, Z., & Wang, J., "Resource Pricing and Allocation in MEC Enabled Blockchain Systems: An A3C Deep Reinforcement Learning Approach," *IEEE Transactions on Network Science and Engineering*, 2021.

[41] Xie, S., Chu, X., Zheng, M., & Liu, C., "A composite learning method for multi-ship collision avoidance based on reinforcement learning and inverse control," *Neurocomputing*, Vol.411, pp.375-392, 2020.

[42] Aggarwal, M., Arora, A., Sodhani, S., & Krishnamurthy, B., "Improving search through A3C reinforcement learning based conversational agent," *International Conference on Computational Science, Springer*, pp.273-286, June 2018.

[43] Wang, B., Zhao, D., Li, C., & Dai, Y., "Design and implementation of an adaptive cruise control system based on supervised actor-critic learning," *2015 5th International Conference on Information Science and Technology (ICIST), IEEE*, pp.243-248, April 2015.

[44] Alfakih, T., Hassan, M. M., Gumaei, A., Savaglio, C., & Fortino, G., "Task offloading and resource allocation for mobile edge computing by deep reinforcement learning based on SARSA," *IEEE Access*, Vol.8, pp.54074-54084, 2020.

[45] Santos, M., López, V., & Botella, G., "Dyna-H: A heuristic planning reinforcement learning algorithm applied to role-playing game strategy decision systems," *Knowledge-Based Systems*, Vol.32, pp.28-36, 2012.

[46] Dong, Hao; Ding, Zihan; Zhang, Shanghang, "Deep Reinforcement Learning (Fundamentals, Research and Applications)," *Springer Singapore*, vol. 10.1007, pp.489-514, 2020.