International Journal of Science and Research (IJSR)
ISSN: 2319-7064
SJIF (2022): 7.942

Unravelling the Complexity: Understanding the Challenges of Reinforcement Learning

Brahmaleen Kaur Sidhu

Department of Computer Science and Engineering, Punjabi University, Patiala, Punjab, India
Email: *brahmaleen.ce[at]pbi.ac.in*

Abstract: ***After an extensive research and exploration of supervised, unsupervised and semi-supervised machine learning algorithms, researchers across the numerous application domains of machine learning are now looking to implement reinforcement learning techniques as they promise a realization of more human-like intelligence in machines. This paper presents a comprehensive body of knowledge about the complexities and challenges that researchers might face while developing reinforcement learning models as solutions for real-life problems. Also, some recommendations have been made in order to assist effective implementation of reinforcement learning algorithms.***

Keywords: computational complexity, environment specification, exploration-exploitation, reinforcement learning, safeRL, sample efficiency

1. Introduction

Creation of computer programs capable of demonstrating intelligence is the main perusal of artificial intelligence. Traditionally, any piece of software that displays cognitive abilities such as perception, search, planning, and learning is considered part of artificial intelligence. Some examples of functionality produced by artificial intelligence software are: pages returned by a search engine, route produced by a GPS app, voice recognition and the synthetic voice of a smart-assistant software, recommended products shown on e-commerce sites, follow-me feature in drones, etc.

Reinforcement learning is a subfield of artificial intelligence and machine learning that focuses on decision making. It is a type of machine learning where an agent learns to make decisions by performing actions and observing the rewards it receives. The goal is to maximize the cumulative reward over time. Basically, the agent learns to make decisions through trial and error. Agent's behaviour is primarily shaped by reinforcement rather than free-will. Positive reinforcement is the strengthening of behaviour by the occurrence of some event (e.g., praise after some behaviour is performed), whereas negative reinforcement is the strengthening of behaviour by the removal or avoidance of some aversive event (e.g., opening and raising an umbrella over your head on a rainy day is reinforced by the cessation of rain falling on you). Behaviours that result in praise/pleasure tend to repeat, behaviours that result in punishment/pain tend to become extinct. In reinforcement learning, the agent and the environment interact with each other, and the agent's decisions influence the state of the environment and the subsequent reward it receives.

The field of reinforcement learning has seen drastic growth in recent years, driven by advances in research, algorithms, computational resources, and applications. This is evident from the surge in number of patents filed in the subject in recent years as shown in Figure 1. The Lens reports that over 84,000 patents records in reinforcement learning [1]. After an extensive research and exploration of supervised, unsupervised and semi-supervised machine learning algorithms, researchers across the numerous application domains of machine learning are now looking to implement reinforcement learning techniques as they promise a realization of more human-like intelligence in machines. Nevertheless, numerous open problems and challenges are faced while implementing reinforcement learning in real life use-cases.

This paper presents a comprehensive body of knowledge about the complexities and challenges that researchers might face while developing reinforcement learning models as solutions for real-life problems. The paper is divided into six sections. The next section traces the history of development of reinforcement learning briefly. The third section introduces the readers to the reinforcement learning framework. The fourth section discusses the background knowledge and a literature of various types of reinforcement learning algorithms. Fifth section elaborates on the challenges and hurdles in the path of practical reinforcement learning. The paper end with a conclusion and a set of recommendations.

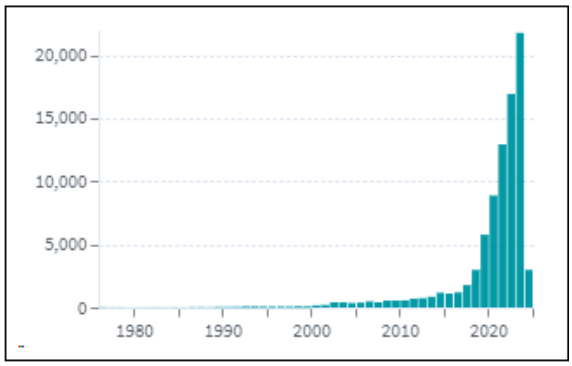

Figure 1: Number of patents published in the field of reinforcement learning in recent years

2. Tracing the Development of Reinforcement Learning

Reinforcement learning was originally inspired by behavioural psychology. The development of reinforcement

learning is believed to be the convergence of three significant paths of study [2]. The first path originated in psychology with Edward Thorndike's influential Law of Effect [3]. Thorndike advanced the notion of associating actions with positive or negative outcomes and examined how individuals learn through a process of trial and error. The Law of Effect is selectional, i.e., it involves trying alternatives and selecting the best by comparing their consequences. Also, Law of Effect is associative, i.e., the alternatives found by selection are associated with particular situations. Thus, the Law of Effect combines search and memory in an elementary form; search in the form of trying and selecting among many actions in every situation, and memory in the form of remembering what actions worked best, associating them with the situations in which they were best. This forms the basis of reinforcement learning.

The second path of study is about the problem of optimal control and its solution using value functions and dynamic programming. The term 'optimal policy' was introduced by Bellman [4] to refer to the most advantageous sequence of decisions according to some preassigned criterion. Bellman argued that the classical approach to the mathematical problems of considering all feasible policies, computing the return from each feasible policy, and then maximizing the return over the set of all feasible policies is not feasible as it will result in an extremely high dimensional space even for a process with moderate number of stages. Quoting Bellman's 'principle of optimality', "An optimal policy has the property that whatever the initial state and initial decisions are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decisions." [4]. He advanced the idea of dynamic programming as that of viewing optimal policy as the one determining the decision required at each time in terms of the current state of the system. Dynamic programming led to the idea of Markovian Decision Processes, an essential component of the theory and algorithms of modern reinforcement learning. Reinforcement learning problems are closely related to optimal control problems.

The third path of study is that of temporal difference learning. Temporal difference learning methods emanate from Minsky's pioneering work of developing methods to simulate behaviour of sentient organisms in computer systems in an understandable way [5]. Minsky's notion of secondary reinforcers being a stimulus that reinforces a behaviour after it has been associated with a primary reinforce forms the basis of temporal learning. Temporal difference learning methods are driven by the difference between temporally successive estimates of the same quantity, such as, the probability of winning a game. Reinforcement learning methods based on temporal learning are particularly effective for learning in environments where decisions are made over time and feedback is delayed. Samuel [6] paved the way for advancements in temporal learning-based reinforcement learning.

Rich Sutton and Andrew Barto, known as founders of the modern field of reinforcement learning, presented adaptive element based on the historical psychological theory of animal learning [7], wherein, reinforcement refers to the strengthening of a pattern of behaviour as a result of the animal receiving a stimulus in an appropriate temporal relationship with another stimulus or response. Behavioural changes produced by reinforcement persist even after the stimulus is withdrawn. Similarly, the adaptive element developed by Sutton and Barto learnt to increase its response rate in anticipation of increased stimulation, thereby producing a conditioned response before the occurrence of the unconditioned stimulus [8].

The three research paths of trial-and-error learning, optimal control, and temporal difference methods were integrated by Watkins [9] in the form of one of the most significant algorithms of reinforcement learning, Q-learning. In the years since, reinforcement learning has witnessed substantial growth, showcasing its versatility in artificial intelligence, machine learning, and other domains. The integration of these diverse viewpoints reflects its collaborative, interdisciplinary approach, continuously shaping and expanding its frontiers.

3. Reinforcement Learning Framework

Reinforcement learning allows an autonomous agent to sense and act in its environment by learning to choose optimal actions to achieve its goals. The goal can be defined by a reward function that assigns a numerical value (an immediate payoff) to each distinct action the agent may take from each distinct state. This reward function may be built into the agent, or known only to an external supervisor who provides the reward value for each action performed by the agent. The task of the agent is to perform sequences of actions, observe their consequences, and learn a control policy. The ideal control policy is one that, from any initial state, chooses actions that maximize the reward accumulated over time by the agent. The reward sent to the agent at any time depends on the agent's current action and the current state of the agent's environment. The agent cannot alter the process that does this. The only way the agent can influence the reward signal is through its actions, which can have a direct effect on reward, or an indirect effect through changing the environment's state.

Figure 2 depicts the working of an agent that is interacting with its environment described by a set of possible states S. The agent can perform any of a set of possible actions A. Each time it performs an action a, in some state s_t the agent receives a real-valued reward r that indicates the immediate value of this state-action transition. This produces a sequence of states s_i, actions a_i, and immediate rewards r_i. The agent's task is to learn a control policy n : S + A, that maximizes the expected sum of these rewards, with future rewards discounted exponentially by their delay.

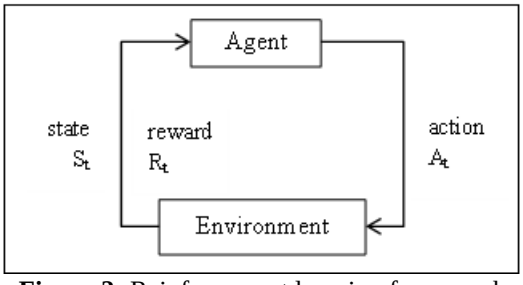

Figure 2: Reinforcement learning framework

The reinforcement learning agent contains two components: a policy and a learning algorithm. A policy defines the learning agent's way of behaving at a given time. A policy is a mapping from perceived states of the environment to actions to be taken when in those states. In some cases the policy may be a simple function or lookup table, whereas in others it may involve extensive computation such as a search process. Within an agent, the policy is implemented by a function approximator with tunable parameters and a specific approximation model, such as a deep neural network. The policy is the core of a reinforcement learning agent in the sense that it alone is sufficient to determine behaviour. In general, policies may be stochastic. The learning algorithm continuously updates the policy parameters based on the actions, observations, and rewards. The goal of the learning algorithm is to find an optimal policy that maximizes the expected cumulative long-term reward received during the task.

4. Literature Review

4.1 Background

Machine learning is the area of artificial intelligence concerned with creating computer programs that can solve problems requiring intelligence by learning from data. There are three main branches of machine learning, namely, supervised learning, unsupervised learning, and reinforcement learning.

Supervised learning is the task of learning from labelled data. In supervised learning, a human decides which data to collect and how to label it. The goal in supervised learning is to generalize. A classic example is a handwritten-digit recognition application; a human gathers images with handwritten digits, labels those images, and trains a model to recognize and classify digits in images correctly. The trained model is expected to generalize and correctly classify handwritten digits in new images.

Unsupervised learning is the task of learning from unlabelled data. Even though data no longer needs labelling, the methods used by the computer to gather data still need to be designed by a human. The goal in unsupervised learning is to compress. A classic example of is a customer segmentation application; a human collects customer data and trains a model to group customers into clusters. These clusters compress the information uncovering underlying relationships in customers.

In supervised learning, the algorithm is trained on a labelled dataset, where each input is associated with a corresponding output or target label. The goal is to learn a mapping from inputs to outputs, enabling the model to make predictions on unseen data. In contrast, unsupervised learning deals with unlabelled data, aiming to discover inherent patterns or structures within the input without explicit guidance. Common tasks include clustering similar data points or reducing the dimensionality of the feature space. While supervised learning emphasizes prediction accuracy through labelled examples, unsupervised learning focuses on revealing the underlying relationships and structures present in the data without predefined labels.

Reinforcement learning can be seen as a way to bridge the gap between artificial intelligence and the natural way that humans and animals learn. Like humans and animals, the agent in reinforcement learning is faced with a series of decisions, and it must choose the action that leads to the highest reward. Reinforcement learning algorithms are designed to learn from their experiences, much like humans and animals. At each step, the agent selects an action based on its current knowledge of the environment and the rewards it has received in the past. This knowledge is stored in a value function, which represents the agent's estimate of the expected reward for each possible action in a given state. The agent then receives a reward from the environment, which is used to update its value function. The updated value function is used to guide the agent's decision-making process in the next step. This process repeats until the agent reaches a terminal state or a stopping condition is reached.

The main advantage of reinforcement learning is its ability to handle complex, uncertain, and changing environments. Unlike supervised learning, reinforcement learning does not rely on a pre-existing dataset, but instead learns from interaction with the environment. This makes it well-suited for applications where the optimal behaviour may change over time, or where it is not possible to explicitly define the desired outcome. Another advantage of reinforcement learning is that it can handle partial observability, where the agent may not have complete information about the state of the environment. This is often the case in real-world applications, such as robotic navigation, where the agent must make decisions based on incomplete or noisy sensor data. To handle partial observability, reinforcement learning algorithms can use techniques such as state abstraction, transfer learning, and deep reinforcement learning. Table 1 highlights the key differences between supervised, unsupervised and reinforcement learning.

Table 1: Key differences between supervised, unsupervised and reinforcement learning

Criteria	*Supervised machine learning*	*Unsupervised machine learning*	*Reinforcement machine learning*
Definition	Learns by using labelled data	Trained using unlabelled data without any guidance	Works on interacting with the environment
Type of data	Labelled data	Unlabelled data	No predefined data
Type of problems	Regression and classification	Association and Clustering	Exploitation or Exploration
Supervision	Extra supervision	No supervision	No supervision
Algorithms	Linear Regression, Logistic Regression, SVM, KNN etc.	K-Means, C-Means, Apriori	Q-Learning, SARSA
Aim	Calculate outcomes	Discover underlying patterns	Learn a series of action
Application	Risk Evaluation, Forecast Sales	Recommendation System, Anomaly Detection	Self-Driving Cars, Gaming, Healthcare

4.2 Reinforcement Learning Algorithms

Reinforcement learning encompasses a variety of algorithms. One of the earliest, Q-learning was introduced by Watkins [9] as a reinforcement learning method of learning to control a Markov Decision Process by incremental dynamic programming. The 'Q' in Q-learning stands for quality that represents how useful a given action is in gaining some future reward. The objective of the model is to find the best course of action given its current state. Q-learning is a model-free, off-policy reinforcement learning that will find the best course of action, given the current state of the agent. Depending on where the agent is in the environment, it will decide the next action to be taken. The learned action-value function, Q, directly approximates the optimal action-value function, independent of the policy being followed. Deep Q-Networks combine Q-learning with neural networks to address complex tasks like multi-robot path planning [10].

SARSA (State-Action-Reward-State-Action) is a model-free, on-policy reinforcement learning method proposed by Rummery and Niranjan [11]. While Q-learning sets the reward for having carried out an action in a state based on the highest-rewarded action available within the new, resulting state, SARSA carries out an second action from the second state according to the policy it has learned and sets the reward for the first state-action pair based on what then happens. Actor-Critic algorithms involve both value-based and policy-based methods to enhance learning [12].

Deep reinforcement learning algorithms use neural networks to represent value functions and policies, carrying an increased ability to handle complex, high-dimensional environments as compared to other algorithms. In another class, policy gradient algorithms, policy is directly parameterized as a probability distribution over actions. These policies are typically represented as neural networks. Policy gradient methods excel in handling stochastic action spaces but can be sample-inefficient. Policy gradient methods are prone to high variance, which can lead to slow convergence and instability in learning. Actor-critic methods are a refinement of policy gradient methods that incorporate elements from both policy-based and value-based reinforcement learning methods. Actor-critic methods introduce a critic component which is typically a value function (e.g., a state-value function or an action-value function) that estimates the expected cumulative reward associated with taking actions in a given state.

5. Challenges

Reinforcement learning is an active and important area of research in artificial intelligence. However, like any scientific discipline, it carries some potential challenges. Reinforcement learning has an inherent complexity. Identifying and addressing these challenges is important for the development of effective algorithms. Despite promising results in the literature, computational complexity, nonstationarity, partial observability, and credit assignment remain significant challenges in this field. Various researchers and developers have reported that reinforcement learning does not work for real-world use-cases.

5.1 Environment Specification

Reinforcement learning deals with sequential decision-making problems where an agent interacts with an environment to achieve a goal. In a typical reinforcement learning problem, there is a learner and a decision maker called agent and the surrounding with which it interacts called the environment (c.f. Figure 2). The environment, in return, provides rewards and a new state based on the actions of the agent. A complete specification of an environment defines a task, one instance of the reinforcement learning problem. So, in reinforcement learning, the agent is not taught how it should do something but is presented with rewards, whether positive or negative, based on its actions. The goal in reinforcement learning is typically to find an optimal policy that maximizes the expected cumulative reward over time.

The reinforcement learning agent and its environment interact at each of a sequence of discrete time steps, t = 0,1,2,… .At each time step t, the agent receives some representation of the environment's state, $S_t \subseteq S$, where S is the set of possible states, and on that basis selects an action, $A_t \subseteq A(S_t)$, where $A(S_t)$ is the set of actions available in state S_t. One time step later, in part as a consequence of its action, the agent receives a numeric reward, $R_{t+1} \subseteq R$, and enters a new state, S_{t+1}. The reward of action A_t is denoted by R_{t+1} instead of R_t because the next reward and next state, R_{t+1} and S_{t+1}, are jointly determined.

Capturing all possible states of the environment effectively for a real-life problem at hand is a big challenge. Real-world environments are often complex and high-dimensional as they usually involve a large number of intrinsic variables. The Designing a state representation that can capture such a high dimensional information can be overwhelming. Also, there is an issue of partial observability or limited perception. The agent might not have access to all the information about the environment making it infeasible to define the complete state.

Also, choosing a highly elaborate state representation might improve the agent's ability to learn complex behaviours, but it can also increase the computational cost of training. Trade-off between accuracy and efficiency is necessary.

Researchers and developers are solely dependent on OpenAI Gym[1], which is an open source Python library for developing and comparing reinforcement learning algorithms and provides a standard set of environments and the required API to communicate between learning algorithms and environments. At the time writing this paper, the environment suite mainly comprises simulated robots and Atari games.

5.2 Computational Complexity

Reinforcement learning agents rely on trial and error while interacting with the environment to learn and thus require large number of interactions with the environment. This is often computationally expensive and time consuming,

[1] https://openai.com/research/openai-gym-beta

especially in real world and complex environments. As discussed in previous sub-section, defining comprehensive state representations often involves numerous features, leading to high dimensionality. Thus, the required training data grows exponentially with the number of features. This exponential growth significantly increases the computational cost of training and can make learning intractable for complex environments.

Deep reinforcement learning algorithms rely on function approximation methods like neural networks to represent value functions or policies. Training these neural networks over the high dimensional state space involves significant computational resources, further contributing to the overall complexity.

5.3 Safety

Efficient reinforcement learning models may be built for toy games and simulation environments, but when it comes to applying reinforcement learning to real-world "safe-critical" tasks such as autonomous driving, ensuring safety becomes a challenge. To exhibit safe behaviour and learn a safe policy that satisfies state-wise safety constraints, the agent needs to evaluate the safety of each state and prevent entering unsafe states. The safety critic that evaluates the safety of the task policy in states and a safety threshold together construct a boundary that divides the state space into safe and unsafe subspaces. Sub-optimal policies may lead to more states being considered as unsafe, thus limiting agent exploration [13]. This leads to a conservative agent since the agent is prone to misjudge under-explored states as unsafe. This greatly limits exploration, which in turn leads to inadequate collection of trajectories to correct the safety critic.

5.4 Complexity of Performance Evaluation

The inherent trial and error nature of reinforcement learning inhibits the usage of general machine learning performance evaluation metrics such as accuracy, precision, recall, f-score, mean squared error and so on. The most fundamental metric that measures the performance of a reinforcement learning agent is the total reward accumulated by the agent over a specific period or episode. Since future rewards are generally less valuable than immediate rewards, a discount factor is often used to give lesser significance to future rewards. Other than this, the percentage of episodes where the agent achieves the desired goal can be indicative of agent's accuracy.

Usually, plotting the agent's average reward over time in the form of a learning curve helps to visualize to identify potential issues like convergence or stagnation. Metrics like exploration rate or entropy of the policy are also used to measure the effectivity of the agent's trade-off between exploration and exploitation. Lower the number of interactions the agent needs with the environment to achieve good performance; more efficient it is considered. Evaluating generalizability of agent to unseen environments is both challenging and important.

Performance evaluation using the above-mentioned metrics is a complex process, because environments of reinforcement learning agents are particularly random and thus, getting a statistically significant evaluation is difficult. Infrequent or delayed rewards call for alternative reward shaping techniques as it may be difficult to attribute success to specific actions. An intuitive method of evaluation is by using ablation studies wherein components of the model or training process are systematically removed and the impact on performance is analysed. Although this helps to identify crucial elements and potential areas for improvement, the process has high computation cost. Running multiple evaluation episodes can also be computationally expensive, especially for complex environments.

5.5 Unpredictability and Inexplainability

Reinforcement learning trade-off between exploring new actions to discover better rewards and exploiting already learned actions that provide known rewards. Unpredictability arises as the agent may prioritize one over the other at different stages of learning. Real-world environments are inherently stochastic, leading to unpredictable outcomes even when the agent follows the same policy and performs same set of actions. Also, the non-linearity in the real-world continuous state spaces can make it challenging to predict the long-term consequences of an agent's actions.

The problem of inexplainability is much prominent in reinforcement learning as compared to other machine learning approaches. First, the high dimensional state space hinders explainability of the agent's decision-making process. Second, in scenarios with delayed rewards or where the outcome is a result of a series of past actions, it becomes challenging to determine which specific action contributed most to the final outcome. This makes it difficult to explain the agent's decision-making process and pinpoint the reasons behind its choices. Third, like other deep learning algorithms, deep reinforcement learning techniques lack transparency of the internal workings of the model.

6. Conclusion and Recommendations

Reinforcement learning comes with numerous challenges. Addressing these challenges is a prerequisite for the development of effective reinforcement learning algorithms. Although the literature portrays a glorious future for reinforcement learning, quite a few implementational hurdles must be crossed to achieve it. Problems like computational complexity both in implementation and evaluation, non- stationarity, partial observability, convergence issues and credit assignment remain largely unsolved. Although deep learning approaches are known to handle high dimensional data well, the problem of sample inefficiency surfaces in deep reinforcement learning as well in addition to other approaches.

Thus, in addition to the general machine learning challenges like requirement of extensive domain knowledge, increased computational complexity in multi-agent environments, increased efficiency requirement in real-time scenarios, the challenges discussed in this paper necessitate the on-going

research and development in reinforcement learning to create more techniques that can learn effectively in distributed modes, with reduced sample complexity, handle high-dimensional states, and operate within real-time constraints. OpenAI's work on Reinforcement Learning with Human Feedback [14], used to train Large Language Models like ChatGPT is a sole and significant milestone in this direction.

Advanced versions of feature engineering and dimensionality reduction techniques need to be incorporated into reinforcement learning models in order to be able to effectively represent the numerous features of environment's state. Clustering may also be used in order to reduce the size of state space such that the state-action mapping may be done effectively.

Explainable reinforcement learning (XRL) techniques is an emerging solution to the "black box" nature of deep reinforcement learning. Model-agnostic methods work with any algorithm and focus on explaining the agent's decisions as to why a specific action was chosen in a given state using techniques like feature attribution and counterfactual explanations. Model-specific methods analysing the parameters of a specific algorithm in order to offer insights into its decision-making process. Learning interpretable policies such as curriculum learning may be used to gradually increase the difficulty of the learning tasks, leading to more predictable behaviour. Study [15] emphasizes the importance of considering the purpose and audience for explanations and suggests focusing on explanations that are actionable and relevant to the specific use case.

As a replacement for deep reinforcement learning models, transformers may also be used. Transformers are a type of neural network architecture that rely on an attention mechanism to understand the relationships between different parts of an input sequence in order to capture long-range dependencies. Transformers consist of encoders that process the input sequence, and decoders that use the encoded information to generate the output sequence. The attention mechanism[2] is used within both encoders and decoders to identify the most relevant parts of the input sequence for a particular concept.

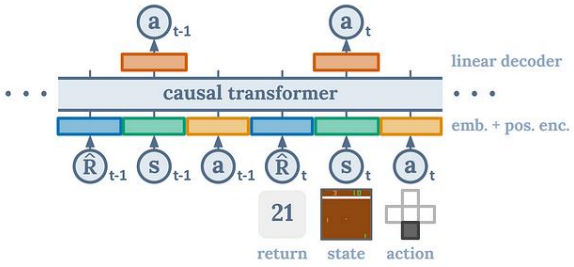

Figure 3: Architecture of Decision Transformer [16]

[2] The attention mechanism allows the model to consider information from any position in the sequence, not just the immediate neighbours. Attention calculations can be performed in parallel for all positions in the sequence, making transformers faster to train compared to sequential processing in other neural network architectures.

Thus, by reformulating a reinforcement learning problem as a sequence-modelling problem, wherein the states, actions, and rewards are laid out in an auto-regressive manner and transformer architecture may be used to find optimal actions. By framing the problem in this way, the Decision Transformer (c.f. Figure 3) can efficiently parse through the sequence of states, actions, and rewards, and intuitively anticipate the best course of action. This results in an algorithm that seamlessly uncovers the most effective strategies, elegantly bypassing the complexities and instabilities often encountered in traditional reinforcement learning methods.

The transformer architecture is reported to be more robust, particularly in situations with sparse or distracting rewards; extremely simple as it requires one network; and matching or even surpassing the state-of-the-art reinforcement learning baselines [16].

Last but not the least, while traditional reinforcement learning emphasizes maximizing reward sans any consideration of safety constraints of the real world environment, a subfield of reinforcement learning called "safe reinforcement learning" focuses on developing algorithms that achieve good performance while ensuring safety [17]. One approach is to modify the objective function to include a penalty term for violating predefined safety constraints. Second is to include human feedback in the form of demonstrations of safe and unsafe behaviours to guide the agent's learning process. Third is to use formal verification techniques to analyse the safety properties of a reinforcement learning gent's policy before deployment.

References

[1] Patent Search Results. The Lens. [Online] [Cited: 1 March 2024.] https://www.lens.org/.
[2] Sutton, Richard S. and Barto, Andrew G. Reinforcement Learning: An Introduction. 2nd. s.l. : The MIT Press, 2015. ISBN-13: 978-0262193986.
[3] A Proof of the Law of Effect. Thorndike, Edward L. 1989, 10 February 1933, Science, Vol. 77, pp. 173-175. DOI: 10.1126/science.77.1989.173.b.
[4] The Theory of Dynamic Programming. Bellman, Richard. 6, 1954, Bulletin of the American Mathematical Society, Vol. 60, pp. 505-515. DOI: S0002-9904-1954-09848-8.
[5] Minsky, Marvin Lee. Theory of Neural-Analog Reinforcement Systems and Its Application to the Brain-Model Problem. s.l. : Princeton Umiversity, 1954. PhD Thesis.
[6] Some Studies in Machine Learning Using the Game of Checkers. Samuel, Arthur Lee. 3, s.l. : IBM, July 1959, IBM Journal of Research and Development, Vol. 3, pp. 210 - 229. DOI: 10.1147/rd.33.0210.
[7] Conditioned Reflexes. Pavlov, Ivan Petrovich. London : Oxford University Press, 1927.
[8] Toward a Modern Theory of Adaptive Networks: Expectation and Prediction. Sutton, Richard S. and Barto, Andrew G. 2, s.l. : American Psychological Association, 1981, Psychological Review, Vol. 88, pp. 135-170.

[9] Watkins, Christopher J. C. H. Learning from Delayed Rewards. Cambridge University. 1989. PhD Thesis.
[10] Multi-robot path planning based on a deep reinforcement learning DQN algorithm. Yang, Yang, Juntao, Li and Lingling, Peng. 3, September 2020, CAAI Transactions on Intelligence Technology, Vol. 5, pp. 177-183. DOI: 10.1049/trit.2020.0024.
[11] Rummery, G. A. and Niranjan, M. On-line Q-learning using connectionist systems. Department of Engineering, University of Cambridge. 1994.
[12] Research on transition state control strategy of propfan engine based on SAC algorithm. Zhou, Jiang-tao, et al. 2023, Journal of Physics: Conference Series, Vol. 2472. DOI: 10.1088/1742-6596/2472/1/012055.
[13] Recovery RL: Safe Reinforcement Learning With Learned Recovery Zones. Thananjeyan, Brijen, et al. 3, s.l. : IEEE, 31 March 2021, IEEE Robotics and Automation Letters, Vol. 6, pp. 4915 - 4922. DOI: 10.1109/LRA.2021.3070252.
[14] Christiano, Paul, et al. Learning from human preferences. OpenAI. [Online] 13 June 2017. [Cited: 4 March 2024.] https://openai.com/research/learning-from-human-preferences. DOI: 10.48550/arXiv.1706.03741.
[15] Finkelstein, Mira , et al. A Survey on Explainable Reinforcement Learning: Concepts, Algorithms, and Challenges. arxiv Preprint. s.l. : arxiv, 30 November 2022. DOI: 10.48550/arXiv.2209.12006.
[16] Decision Transformer: Reinforcement Learning via Sequence Modeling. Chen, Lili, et al. 2021, Advances in neural information processing systems, Vol. 34, pp. 15084-15097.
[17] Zhang, Xiao, et al. Safe Reinforcement Learning with Dead-Ends Avoidance and Recovery. arxiv Preprint. s.l. : arxiv, 24 June 2023. DOI: 10.48550/arXiv.2306.13944.

Author Profile

Dr. Brahmaleen K. Sidhu is an Assistant Professor in the Department of Computer Science and Engineering, Punjabi University, Punjab, India and has around 18 years of teaching experience. She holds a Ph.D. degree in Faculty of Engineering and Technology from Punjabi University, an M.Tech. degree in Computer Science and Engineering from the Punjab Technical University and a B.Tech. degree in Computer Science and Engineering from Punjabi University. Her research interests include software architecture, software evolution, software quality, refactoring, model-driven development, data science and machine learning. Dr. Sidhu has around 75 research papers in international journals and conferences, and a book titled "A Handbook of Reinforcement Learning" published in 2023. She has been awarded the “International Innovative Educator Award 2021" and is listed in “100 Eminent Academicians of 2021” by International Institute of Organized Research.