**International Journal of Engineering Sciences & Emerging Technologies, Oct. 2023.**
**ISSN: 22316604 Volume 11, Issue 2, pp: 34-41 ©IJESET**

# A COMPARATIVE STUDY OF DEEP REINFORCEMENT LEARNING ALGORITHMS IN ROBOTICS

Ameen Ur Rehman

[1]Department of Computer Science and Engineering,
Jamia Hamdard (Hamdard University), New Delhi, India
ameensadiya12345@gmail.com

***ABSTRACT***

*Deep Reinforcement Learning (DRL) is a promising field within Artificial Intelligence that combines deep learning, a subfield of Machine Learning, with Reinforcement Learning (RL) techniques. DRL allows deep neural networks to represent and comprehend the world, enabling agents to acquire proficiency in intricate tasks and make decisions in high-dimensional environments. In this paper, we explore different types of RL approaches, including Model-based, Value-based, and Policy-based methods and combinations of deep neural networks with RL techniques, allowing for effective representation learning from high-dimensional inputs. The paper discusses various deep-reinforcement learning algorithms and applications that are commonly used in robotics and other domains. The paper discussed the selection of a specific RL approach is influenced by factors like problem complexity, the presence of a suitable training environment, and the desired decision-making process. Finally, we conclude by discussing the future challenges and opportunities for deep reinforcement learning in autonomous robotics.*

***KEYWORDS:*** *Robotics, Artificial Intelligence, Reinforcement Learning, Mathematics, Deep Reinforcement Learning.*

## 1. INTRODUCTION

Deep Reinforcement Learning (DRL) stands out as an incredibly captivating domain within the realm of Artificial Intelligence (AI), also the subset of artificial intelligence that combines deep learning which is a specialized area of machine learning that emphasizes the utilization of neural networks to make predictions from the input sample data in a supervised way and encode the representation of that data in the way to get insights from that data. And combines Reinforcement Learning, a type of learning where an agent or an Intelligent system interacts with the environment to learn through rewards and punishment. Rewards and Punishments in the Trial & Error phase of learning is a fundamental process through which reinforcement learning agents learn. Deep Reinforcement Learning (DRL) empowers deep neural networks to effectively understand and interpret the surrounding environment, enabling them to take meaningful actions through an autonomous agent. It also enables agents to learn complex tasks using neural networks and make decisions in environments with high dimensional input spaces [1][2][4]. In traditional Reinforcement Learning the agent at each step in the process first executes the action, and observes a new state, and the feedback an RL agent receives, consisting of rewards and penalties, plays a crucial role in its learning process. The significance of life for reinforcement learning agents is to maximize the accumulative reward via Policy, Value functions. **Policy function**: An agent operates based on a strategy called the Policy. It sees the world and makes a decision that's a policy that decides how to act. Reinforcement learning aims to disclose an optimal policy that maximizes the predicted cumulative reward. **Value** function: It is a function that assigns a value to each individual state or combination of a state and action pair in an environment. It is essential for both policy evaluation and value improvement in reinforcement to maximize its long-term rewards. Policy functions define the agent's strategy for decision-making, while value functions assign values to states or state-action

DOI: 10.5281/zenodo.10434199

pairs and are important for policy evaluation and improvement. Policy and value functions are fundamental components extensively examined in the subsequent paragraphs. These components play a significant role in reinforcement learning [3][4][5][11][13][14].

## 2. TYPES OF REINFORCEMENT LEARNING (RL)

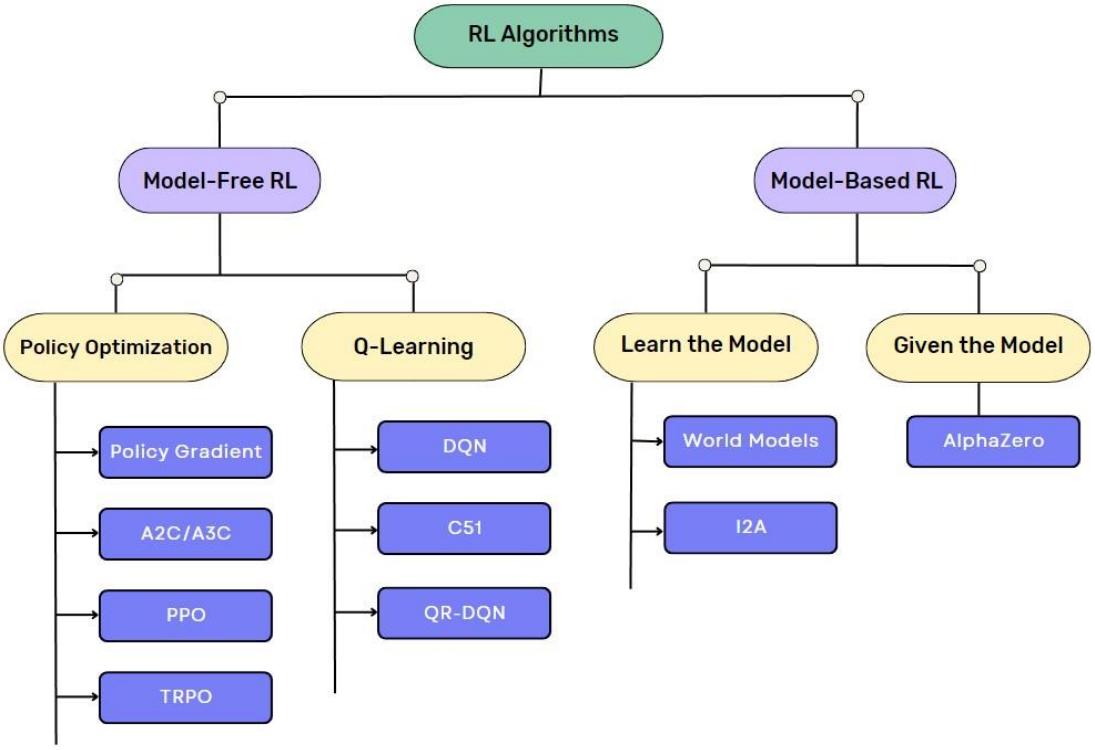


**Figure 1.** Taxonomy of Reinforcement Learning

### 2.1. Model Based

Model-based reinforcement learning is an approach that involves understanding a representation of the environment without engaging in direct interaction. The agent initially learns an approximate dynamics model and frequently updates and re-plans it. This method can be utilized for planning, exploration, and policy optimization, enabling efficient learning without relying solely on real-world interactions.[2][3] The key steps involved in model-based RL are: ***i) Learning of Model*:** The agents gather data by interacting with the environment and this data is used to find insights and patterns. The model can be represented as a function that takes the current state and actions as input and expects the reward and next state. ***ii) Policy Optimization*:** The learned model can be used for planning and policy optimization. Planning involves using the model to simulate possible sequences of actions and states, allowing the agent to evaluate potential future trajectories and make informed decisions. Policy optimization techniques such as value iteration, and stochastic optimization can be applied using the learned model to improve the agent's policy. ***iii) Exploration:*** The learned model can also be used for exploration. By simulating possible actions and their outcomes using the model, the agent can explore and gather valuable information about the environment without directly interacting with it. This exploration based on the model can be more sample-efficient and safer than exploration in the real environment. ***iv) Model Refinement:*** As the agent interacts with the environment based on the learned model, it continues to collect new data and refines the model iteratively. This process allows the model to improve its accuracy over time, leading to more accurate planning and policy optimization [2][4][7][8].

DOI: 10.5281/zenodo.10434199

Applications of model-based reinforcement learning include ***a) Robotics:*** Model based reinforcement learning has been applied to robotic control tasks, such as robot arm manipulation, locomotion, and object grasping. By learning and leveraging a model of the robot's dynamics, the agent can plan and optimize its control policy efficiently. ***b) Autonomous Systems****:* Model based RL can be used in autonomous systems, such as autonomous driving or unmanned aerial vehicles (UAVs). By acquiring knowledge about the environment, the agent can strategize and improve its decision-making, resulting in enhanced efficiency and safety. ***c) Game Playing:*** Model-based reinforcement learning has demonstrated successful applications in playing games, both in simulated environments and real-world situations. By acquiring a model of the game dynamics, agents can simulate and strategize future moves, empowering them to make well-informed decisions [3][4][9][10].

Model based reinforcement learning offers several advantages, including improved sample efficiency, safety, and the ability to perform offline planning. However, it also introduces the challenge of accurately learning and representing the environmental dynamics. Properly addressing this challenge is crucial for the success of model-based reinforcement learning approaches.

## 2.2. Value based

It is an approach in which an agent learns to assess the value of states or state-action pairs without explicitly learning a policy. The value function is used to evaluate the desirability or quality of different states or state-action pairs, guiding the agent's decision-making process. In value-based RL, the objective of the agent is to acquire knowledge about either the state value function (V-function) or the action-value function (Q-function). The state value function, referred to as *V(s),* provides an estimate of the anticipated total reward when starting from a given state, s. Conversely, the action-value function, denoted as *Q (s, a),* gives an approximation of the total expected reward when taking action "*a*" from state "*s*" and following a predetermined policy thereafter. Here are the key steps involved in value-based reinforcement learning: ***i) Value Function Estimation:*** The agent iteratively updates its estimates of the value function based on observed rewards and the estimated values of future states or state-action pairs. This estimation can be done using techniques like temporal difference (TD) learning, which updates the value function based on the difference between predicted and actual rewards***. ii) Policy Improvement:*** After estimating the value function, the agent can improve its policy by selecting actions that have higher estimated values according to the value function. This process is known as policy improvement and can be done by following a greedy policy concerning the value function estimates (i.e., selecting the action with the highest value). ***iii) Balancing Exploration and Exploitation:*** Striking the optimal balance between exploration and exploitation plays a crucial role in value-based reinforcement learning. Initially, the agent explores the environment to gather reward and transition information. As learning advances, it gradually transitions to exploiting the learned value function to make optimal decisions. ***iv) Value Iteration or Q-learning:*** Value-based methods often use iterative algorithms like value iteration or Q-learning to estimate and update the value function. These algorithms iteratively update the value estimates based on the observed rewards and transitions, converging toward the optimal value function [2][5][6].

Applications of value-based reinforcement learning include: ***a) Game Playing*****:** Value-based methods, such as Q-learning, have demonstrated their effectiveness in game-playing scenarios, encompassing various types of games ranging from traditional board games to intricate ones like Go. These techniques enable the agent to assess the value associated with distinct game states or state-action combinations, allowing it to effectively strategize and make informed decisions. ***b) Control in Continuous Spaces*****:** Value-based RL is also used in control problems involving continuous state and action spaces. Techniques like, Deep Q-Networks (DQN) which is a method that leverages the power of deep neural networks to effectively address complex and high-dimensional state spaces within the context of reinforcement learning, allowing for control in domains such as robotics or autonomous systems. ***c) Resource Management:*** Value-based RL has practical applications in resource management problems. In this context, the agent learns to assess the value of various actions or decisions, enabling efficient optimization of resource allocation, scheduling, and routing tasks [4].

DOI: 10.5281/zenodo.10434199

Value-based reinforcement learning has proven to be effective in various domains, particularly when the focus is on estimating the value function rather than directly learning a policy. It offers advantages such as simplicity, scalability, and the ability to handle high-dimensional state spaces. However, it may face challenges when dealing with continuous or complex action spaces and in cases where a policy needs to be explicitly learned.

### 2.3. Policy based

Author One approach involves training an agent to learn a policy, which acts as a guide for selecting actions based on given states, without the need for explicit estimation of value functions. In policy-based reinforcement learning, the agent aims to directly learn a parameterized policy function $\pi(a \mid s)$ that specifies the probability or deterministically selects an action in a given state, *s*. The policy can have two types: stochastic and deterministic. In the stochastic policy, it provides a probability distribution for each state, indicating the likelihood of taking different actions. On the other hand, the deterministic policy directly maps states to specific actions without any probability distribution involved. Here are the key steps involved in policy-based reinforcement learning: ***i) Policy Parameterization*:** The policy function is typically parameterized using a function approximator, such as a neural network, which takes the state as input and outputs the action probabilities or deterministically selects an action. ***ii) Policy Evaluation:*** The agent collects data by interacting with the environment using the current policy. The collected data is then used to estimate the performance of the policy through techniques such as Monte Carlo sampling or temporal difference learning. ***iii) Policy Optimization:*** The estimated performance of the policy is used to update the policy parameters to improve its performance. This is typically done using gradient-based optimization methods, such as stochastic gradient descent, policy gradients, or natural gradient methods. ***iv) Balancing Exploration and Exploitation:*** Balancing exploration and exploitation is crucial in policy-based reinforcement learning. The agent needs to explore the environment to discover better policies but also needs to exploit the learned policy to maximize rewards [2][5][6][13][15]. Exploration is often achieved by introducing randomness or by using exploration strategies like epsilon-greedy or adding noise to action selection.

Applications of policy-based reinforcement learning include: ***a) Continuous Control*:** Policy-based methods are well-suited for continuous control problems where the action space is continuous, such as robotic manipulation, locomotion, or autonomous vehicle control. The policy can directly output continuous actions without the need for discretization. ***b) High-Dimensional State Spaces*:** Policy-based methods, particularly those utilizing deep neural networks, can handle high-dimensional state spaces effectively. They have been successfully applied in tasks such as image-based control, natural language processing, and computer vision. ***c) Multi-Agent Systems*:** Policy-based RL is also applicable in multi-agent systems where multiple agents interact and learn policies simultaneously. It allows for learning cooperative or competitive behaviors in complex environments [4][5].

Policy-based reinforcement learning offers advantages such as the ability to handle continuous action spaces, flexibility in representing complex policies, and direct optimization of the policy objective. However, it may require more data and training time compared to value-based methods, and it can be sensitive to the choice of policy parameterization and optimization techniques. So, all these types are crucial to understanding Deep Reinforcement Learning accurately and using its algorithms in robotics. Now let's deep dive into deep reinforcement Learning and its algorithms in robotics [5].

## 3. DEEP REINFORCEMENT LEARNING

Deep reinforcement learning combines deep neural networks with reinforcement learning techniques, allowing for effective representation learning from high-dimensional inputs. Deep neural networks enable value function estimation, policy parameterization, and function approximation in both value-based and policy-based methods. In deep reinforcement learning, Model based, Value based, and Policy based approaches can be combined or used interchangeably depending on the problem domain and the specific requirements. The choice of approach depends on factors such as the environment's nature, the

DOI: 10.5281/zenodo.10434199

task's complexity, the available data, the desired trade-offs between exploration and exploitation, samples, and computational efficiency [6][13][14][16].

In the field of deep reinforcement learning, algorithms have shown promising results in tackling diverse robotics tasks by leveraging the remarkable capability of neural networks to create meaningful representations from exceptionally intricate and multidimensional sensory data. The raw data is collected from sensors present in the machine or robot's sensors can be cameras, radar, IMU, audio, GPS, etc. to learn agents for the real-world environment and to achieve high-performance control.

## 4. DEEP REINFORCEMENT LEARNING ALGORITHMS IN ROBOTICS

Deep reinforcement learning algorithms have shown promise in enabling robots to discover complex skills and acclimate to new environments. These algorithms use deep neural networks to approximate value and policy functions, but challenges such as exploration, generalization, and data collection must be addressed to make them more effective for real-world robotic applications.

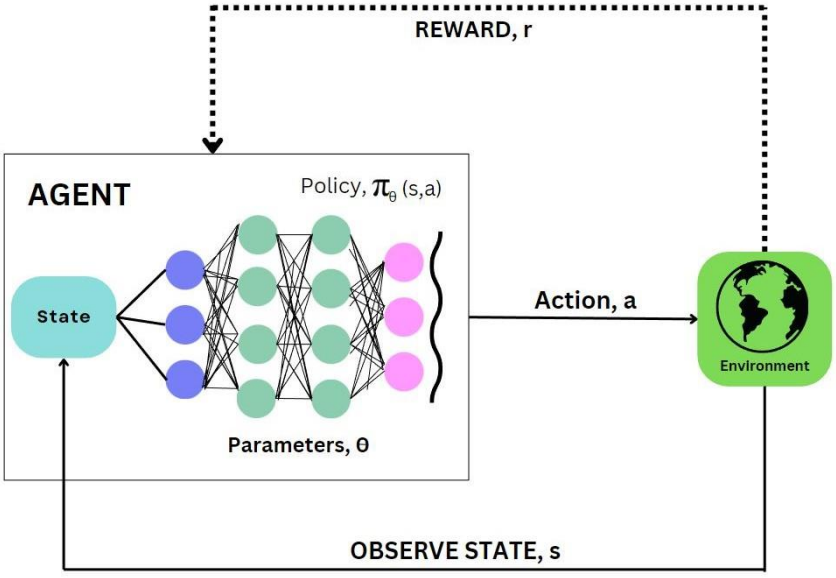


**Figure 2.** Deep Reinforcement Learning Model Cycle Learning

### 4.1. Deep Q-Learning

The algorithm described here combines aspects of deep learning and reinforcement learning to form a model-free approach. Its aim is to enable an agent to realize the best actions in a given scenario. The primary objective of reinforcement learning is to find an optimal policy that maximizes the agent's total rewards within the environment. This is achieved through the use of Q-learning, where Q-values represent the expected future rewards an agent would receive for taking specific actions in particular states. Through iterations, the Q-learning algorithm continually adjusts these Q-values based on the agent's experiences gained from interacting with the environment. Deep Q-learning introduces a deep neural network known as a Q-network. This network takes the current state of the environment as input and generates corresponding Q-values for each available action. During training, the Q-network is optimized to minimize the discrepancy between predicted Q-values and target Q-values, which are updated using the Q-learning update rule. Deep Q-learning has been successfully applied to a vast range of tasks, including playing video games, controlling robots, and optimizing resource allocation. Its ability to learn from raw sensory inputs makes it a powerful algorithm for solving complex and high-dimensional problems [8][14][15].

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha(R_{t+1} + \gamma max_a Q_t(s_{t+1}, a_t) - Q_t(s_t, a_t) \qquad -(i)$$

DOI: 10.5281/zenodo.10434199

$Q_{t+1}(s_t, a_t) = New\ State,\ \ Q_t(s_t, a_t) = Old\ State, R_{t+1} = Reward, \alpha = Learning\ Rate, \gamma = Discount\ Factor$

**Loss Function (Squared Error):** The loss **represents the error between observation and expectation**; a differentiable loss function is needed to properly perform the update. For deep Q-learning, the loss function is typically a simple mean-squared error. The loss function in Q-learning provides a measure of the discrepancy between predicted and target Q-values, guiding the learning process to optimize the Q-function and improve the agent's decision-making capabilities in an overall Deep reinforcement learning setting.

$$L = E[(r + \gamma max_{a`} Q(s`, a`) - Q(s, a))^2 \ ] \quad -(ii)$$

$$r + \gamma max_{a`} Q(s`, a`) = Target, Q(s, a))^2 = Prediction$$

### 4.2. Proximal Policy Optimization (PPO):

PPO is a reinforcement learning approach that diminishes the computational burden corresponding to TRPO. PPO combines the restraint into the objective function by penalizing the KL divergence between old and new policies, multiplied by a constant C. This allows for uncomplicated optimization utilizing stochastic gradient descent. An important contemplation in PPO is the proper selection of the constant value C. PPO has demonstrated effectiveness across various tasks, such as robotic manipulation, locomotion, and game-playing [17].

### 4.4. Trust Region Policy Optimization (TRPO)

TRPO is a policy optimization algorithm that aims to maximize the expected reward while staying within a trust region of policy changes. It ensures that policy updates are performed conservatively, maintaining a close approximation to the previous policy to guarantee monotonic improvements [7][15][17]. TRPO is known for its robustness and safety guarantees.

### 4.3. Advantage Actor-Critic (A2C)

A2C (Advantage Actor-Critic) is a reinforcement learning algorithm that integrates policy-based and value-based methods. The A2C algorithm utilizes a deep neural network to learn a policy (actor) and simultaneously estimate the state value function (critic). A2C aims to enhance learning stability and efficiency by combining policy gradients and value functions [15][17][18].

### 4.5. Asynchronous Advantage Actor-Critic (A3C)

A3C is a parallelized version of the A2C algorithm. It utilizes multiple actor-learner agents that interact with multiple instances of the environment in parallel. This parallelization allows for more efficient exploration and faster convergence. A3C has been successful in learning policies for complex and challenging tasks [17][18].

### 4.6. Deep Deterministic Policy Gradient (DDPG)

DDPG is a well-known off-policy actor-critic algorithm that is tailored to handle continuous action spaces. Its architecture incorporates deterministic policy gradients and utilizes deep neural networks to acquire a deterministic policy while estimating the action-value function. To ensure stable learning, [17][18] DDPG leverages experience replay and employ target networks, which bear resemblance to the mechanism employed in DQN.

Here are some other examples of deep reinforcement learning algorithms, including Soft Actor-Critic *(SAC),* Proximal Deterministic Policy Optimization *(PDPO),* and Distributed Distributional Deterministic Policy Gradients *(D4PG).* Each algorithm has its own unique strengths and applications. The selection of an algorithm depends on factors such as the problem domain, the characteristics of the environment, and the desired balance between exploration, exploitation, stability, and efficiency [13][14][15][17][18].

DOI: 10.5281/zenodo.10434199

## 5. CHALLENGES

Deep reinforcement learning in robotics faces several challenges, including: ***a) Exploration*:** Exploration is a major challenge in robotic reinforcement learning. Robots need to explore their environment to learn optimal policies, but exploration can be time-consuming and potentially unsafe. Various techniques, such as utilizing demonstrations or hand-engineered controllers, have been proposed to address this challenge. ***b) Generalization***: Generalization is another challenge in deep reinforcement learning. While deep neural networks have the ability to generalize from training data, it can be difficult for RL policies to generalize to unseen situations. However, the presence of extensive and varied data can empower RL policies to generalize akin to supervised models. ***c) Data Collection*:** Acquiring advanced expertise in robotics necessitates significant data gathering from the robots in order to master intricate skills. This process often requires keeping the robots operational with minimal human intervention. Ensuring the robots can collect sufficient and diverse data is a challenge in itself. ***d) Sample Efficiency*:** Deep reinforcement learning algorithms often suffer from sample inefficiency, which refers to their requirement for a significant number of interactions with the environment to discover optimal policies. This characteristic can pose challenges in real-world robotic applications due to the associated costs and time constraints. ***e) Safety and Risk*:** Reinforcement learning algorithms need to ensure safe behavior during learning and deployment. Preventing unsafe actions and defining appropriate reward functions to guide learning are important challenges in robotic reinforcement learning.

Overcoming these challenges necessitates continuous exploration and the creation of innovative algorithms and methods to enhance the effectiveness and feasibility of deep reinforcement learning in the field of robotics.

## 6. CONCLUSION

In conclusion, deep reinforcement learning has shown great promise in robotics, with successful applications in tasks like walking on four legs, picking up unfamiliar objects, and mastering intricate movements. However, several challenges still need to be addressed to make deep RL more effective and practical for real-world robotic applications. These challenges include exploration, generalization, data collection, sample efficiency, safety and risk, and real-world constraints. Future work in this area should focus on developing new algorithms and techniques that can address these challenges. For example, developing more efficient exploration strategies, improving generalization capabilities, and developing methods for safe and efficient data collection could all help to make deep RL more effective in robotics. Additionally, developing algorithms that can work within the constraints of real-world robotic systems, such as limited computational resources and sensor noise, will be important for practical applications. Overall, Deep reinforcement learning can transform robotics by enabling robots to learn complex skills and adapt to new environments, but further progress is needed to achieve this goal. By addressing the challenges outlined above and continuing to develop new algorithms and techniques, we can unlock the full potential of deep RL in robotics and create robots that are more capable, adaptable, and intelligent.

## REFERENCES

[1] Lee, Berk Altuner, A., Kilimci, Z. H., (2022). "A Novel Deep Reinforcement Learning-Based Stock Price Prediction Using Knowledge Graph and Community-Aware Sentiments."

[2] Han, D., Mulyana, B., Stankovic, V., & Cheng, S. (2023). "A Survey on Deep Reinforcement Learning Algorithms for Robotic Manipulation."

[3] Ibarz, J., Tan, J., Finn, C., Kalakrishnan, M., Pastor, P., & Levine, S. (2020). "How to Train Your Robot with Deep Reinforcement Learning – Lessons We've Learned". SAGE. DOI: 10.1177

[4] Shadow Robot. (2021) "How AI and Machine Learning Can Improve Robotics."

[5] Russell, S. J., & Norvig, P. (2010). *Artificial Intelligence: A Modern Approach (3rd ed.)*. Pearson Education.

DOI: 10.5281/zenodo.10434199

[6] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., & Wierstra, D. (2015). "Continuous control with deep reinforcement learning." arXiv preprint arXiv:1509.02971.

[7] Schulman, J., Levine, S., Moritz, P., Jordan, M. I., & Abbeel, P. (2015). "Trust region policy optimization". Proceedings of the 32nd International Conference on Machine Learning (ICML), 1899-1907.

[8] Van Hasselt, H., Guez, A., & Silver, D. (2015). "Deep reinforcement learning with double Q-learning". arXiv preprint arXiv:1509.06461.

[9] Tian, Z., Pattanaik, A., Liu, S., Bommannan, G., & Chowdhary, G. (2018). "Robust deep reinforcement learning." arXiv preprint arXiv:1803.07295.

[10] Kulkarni, T., Narasimhan, K., Sastry, S., & Tenenbaum, J. (2016)." Hierarchical reinforcement learning for robotic control." arXiv preprint arXiv:1604.06772.

[11] Tian, Z., Pattanaik, A., Liu, S., Bommannan, G., & Chowdhary, G. (2019). Bayesian robust deep reinforcement learning for robotic control. IEEE Transactions on Robotics, 35(6), 1746-1758.

[12] Carta, Salvatore, Andrea Corriga, Anselmo Ferreira, Alessandro Sebastian Podda, and Diego Reforgiato Recupero (2021)." A multi-layer and multi-ensemble stock trader using deep learning and deep reinforcement learning".Applied Intelligence, 51(2):889-905.

[13] Aboussalah, Amine Mohamed, and Chi-Guhn Lee (2020)." Continuous control with stacked deep dynamic recurrent reinforcement learning for portfolio optimization." Expert Systems with Applications,140: 112891.

[14] Liang, Zhipeng, Hao Chen, Junhao Zhu, Kangkang Jiang, and Yanran Li (2018)." Adversarial deep reinforcement learning in portfolio management." arXiv preprint arXiv:1808.09940.

[15] Spooner T., Fearnley J. Savani R. and Koukorinis (2018). "A. Market making via reinforcement learning" arXiv preprint arXiv:1804.04216.

[16] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M. (2015)." Human-level control through deep reinforcement learning." Nature, 518(7540), 529-533.

[17] Schulman, J., Levine, S., Abbeel, P., Jordan, M. I., & Moritz, P. (2017). "Proximal policy optimization algorithms." arXiv preprint arXiv:1707.06347.

[18] Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018). "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor". International Conference on Machine Learning (ICML).

DOI: 10.5281/zenodo.10434199