Journal of Theoretical and Applied Information Technology
28th February 2022. Vol.100. No 4
2022 Little Lion Scientific
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

DEEP REINFORCEMENT LEARNING TO MULTI-AGENT DEEP REINFORCEMENT LEARNING

MEHDI SAMIEIYEGANEH [1], PROFESSOR RAHMITA WIRZA BT O. K. RAHMAT [2], DR FATIMAH BINTI KHALID [3], DR KHAIRUL AZHAR BIN KASMIRAN[4]

[1]Faculty of Computer Science and Information Technology, University Putra Malaysia (UPM), Serdang, MALAYSIA,
[2]Faculty of Computer Science and Information Technology, University Putra Malaysia (UPM), Serdang, MALAYSIA,
[3] Faculty of Computer Science and Information Technology, University Putra Malaysia (UPM), Serdang, MALAYSIA,
[4] Faculty of Computer Science and Information Technology, University Putra Malaysia (UPM), Serdang, MALAYSIA,
E-mail: [1]gs51922@student.upm.edu.my, [2]rahmita@upm.edu.my, [3]fatimahk@upm.edu.my ,
[4]k_azhar@upm.edu.my

ABSTRACT

Machine Learning (ML) has been a remarkable success in the last few years, which Reinforcement Learning (RL) has seen rapid growth with new techniques that have revolutionized the area. Sequential -Decision Making tasks are a main topic in ML, these are tasks based on deciding, the sequence of actions from experience carry out in an environment that is uncertain to achieve goals In this paper, we discuss topics such as Deep Learning (DL) and Multi-agent Systems (MAS) that are used in RL as Deep Reinforcement Learning (DRL) and Multi –Agent Deep Reinforcement Learning (MADRL). In fact, overall goal in this paper is a comprehensive explanation of the various Deep Reinforcement Learning (DRL) algorithms, and its combination with Multi-Agent methods. To achieve this goal, in section 2, we have reviewed the articles that are the founders of these methods and have also used various methods in the field of MADRL. In the third section, we look at the RL and important algorithms that exist in this area. In the fourth section, we study DRL and explain the reasons for which different algorithms have been developed in this regard. In the fifth section, we will look at the MADRL and address some of the challenges and work that has been done in this area..At the end of this section we mentioned some important papers in the table with their methods, which is used. The sixth section provides an explanation of the research currently being done by the authors, as well as interesting topics for researchers to use in future research. Given that we have tried to explain the concepts in a simple and straightforward way in this paper, we hope that the materials mentioned are suitable for novice researchers in this field.

Keywords: *Machine Learning; Reinforcement Learning; Deep Learning; Deep Reinforcement Learning; Multi-Agent Systems; Multi –Agent Deep Reinforcement Learning.*

1. INTRODUCTION

At the beginning of this article, a brief explanation is given about the Reinforcement Learning and the works done on this subject, which is useful for understanding the concepts. Trial and Error (TE) is a process in which RL operates based on that. By interacting, that means taking actions within an environment the agent or learner will be able to learn, this process done by Thorndike in 1986[1]. in 1954, Stochastic Neural-Analog Reinforcement Calculator (SNARCs) computer, designed by Minsky, that was first computer based on neural. The work of This PC was based on simulation of rat's brain to explain the puzzle of maze, nearly after two decades the TE learning

based on model of computational and Temporal-Difference (TD) learning strategies dependent on psychology unified by Klopf. In 1989, it's time for Q-learning to emerge, the Q-learning formed with fetched optimal control[2] by Watkins alongside Dayan that done based on Markov Decision Process (MDP) and bellman equation as well as TD learning. Q-learning was good and used to solve so many problems in the world but there are also problems for example it cannot applied to high-dimensional issues because when number on inputs are increasing the calculations also will be increased. Furthermore one of main drawbacks in RL is the need for defining (hand-crafting) features used to learn (i.e. curse of dimensionality and overpass the computational constraint of conventional PCs)[3], In 2015 the problem of curse of dimensionality partially solved by Mnih et al. [4] their success was based on combining DL with RL. DRL turned into a standardizing approach in AI and has been highly studied by research groups. Achievements of the improvement of RL are exhibited in (Fig. 1) that range from the experimentation technique to DRL [3].

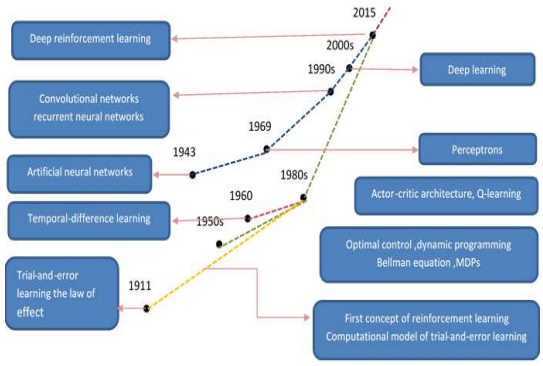


Fig. 1. Achievements of the improvement of RL

Reinforcement Learning can simulate human learning capacity to choose activities that enlarge long –term benefit in their communications with environment. One of the differences that Reinforcement Learning gain with other learning methods i.e. Supervised and Unsupervised Learning, is use of the agent. So, Reinforcement Learning can be utilized to build up an agent (Learner), the agent is tantamount to the human execution. For example, in Mahadevan and Connell[5] , Schaal [6], Benbrahim and Franklin [7], Riedmiller et al. [8] and Muelling et al.[9], Reinforcement Learning existed broadly utilized in mechanical autonomy and robotics frameworks. For a better understanding of various types of machine learning methods, we provide a general definition of the differences between machine learning methods, actually in Supervised Learning we do learning by information as input and corresponding output by an external supervisor that frequently named "labelled data", though RL is learning by communicating through the obscure condition. And furthermore, unsupervised learning will be figuring out how to investigate the concealed structure of information where data is obscure ("unlabelled data"). RL is a goal directed learning process and the goal of RL is to obtain the maximum reward [3]. Complementing each other, DR pulse RL generated a new area that named DRL (Deep Reinforcement Learning) [10].

The achievement of DRL in 2015 when Mnih et al. [4] is really marked by modern RL, made utilization of Deep Q-Network (DQN) [3]. DQN building the agent that in compare with a specialist player in a movement of 49 extraordinary Atari preoccupations achieves the best outperform [11]. OpenAI in 2017, introduced a robot, and announced that the robot could defeat capable player in web game Dota 2, which ought to be more caught than the Go game. These destinies give the essential stimulus to big business companies, for example, Tesla, and Uber because they are having a competition to create self-driving vehicles based on self-driving.

DRL recognizes with DL strategy. For learn a problem DL utilizes Multi-layer neural networks to in various dimensions of abstraction [12]. To manage high-dimensional information DRL use DL as an approximation function. This reality makes DRL a promising way to deal with complex issues and solve them [3]. As mentioned, one of the most important features of DL is the use of Neural Networks (NNs) that can find compact representations in high- dimensional data [13], thus eliminating the need for manual feature design. Deep learning has many uses in the real-world problems. For example, in medical image processing, it has been used extensively. In order to better study this, we recommend readers' resources to[14],[15] ,[16]. Deep NNs are trained to estimate the optimal value function or policy or both of them, where the promise of generalization is expected to be delivered by the representation ability of deep NNs (as the function approximator). One of the key advantages of DRL is that it enables RL to scale to conditions that include high-dimensional states and actions.

Given that real-world issues are growing and complicated, therefore, in some cases, a single agent will not be able to adapt to complex problems, so, the utilizations of a Multi-Agent System (MAS) are essential [17].Very nearly 20 years prior original review by Stone and Veloso's [18] laid the foundation for characterizing the

domain of MAS (Multi-Agent Systems) and its open issues in AI. Around ten years back, Grenager and Shoham [19] wrote the literature based on Multi-agent Learning (MAL) that it was not includes all related papers [3]. Since then, the quantity of distributed MAL works proceeds to steadily rise, which led to different surveys on the area, ranging from analyzing the fundamentals of MAL with challenges in this area, [20],[21], [10], to tending to explicit subareas: game theory and MAL [19],[22], cooperative scenarios[23],[24], and evolutionary dynamics of MAL [25]. In just the last couple of years, two surveys related to MAL have been published: learning in non-stationary environments[26], [3] and agents modelling agents[27], [3]. Two major successes in MAL was as follows [17], first, in single-agent computer games [4]; and as well in games with two-player, e.g., playing Go [28],[29], poker [30], and games of two competing teams, e.g., DOTA 2 . To maximize the result the agents in a MAS must cooperate or compete with each other [17]. For example, competitive and cooperative robots, unmanned aerial vehicles, traffic control system and online games based on multi-player can be mentioned in this area. Between numerous utilizations of DRL in the writing, there are an enormous number of studies utilizing DRL in MAS, which named Multi –Agent Deep Reinforcement Learning (MADRL). Due to developing a single agent space to a Multi-agent condition makes a few difficulties [3]. for instance, In the field of stability and adaptability, studies have been carried out by Busoniu et al. [10], Also in the field of evolutionary dynamics, studies have been conducted by Bloembergen et al. [25], Hernandez-Leal et al. [17] Have conducted studies on learning to cooperate and the emergence of behavioral factors among agents, and Silva et al.[31] investigated strategies for learning reuse autonomy in MARL [3]. Multi-agent learning is much more difficult than learning in single agent, and the reason for this can be found in topics like non-stationary, curse of dimensionality, and Multi-agent credit assignment[19],[10],[26],[32],[33].Despite this complexity, works reporting successes in MDRL published in various AI conferences [3]. This paper contributes to the state of the art with a brief survey of the current works in MDRL in an effort to complement existing surveys [26], [27],[13],[3], [17] and presents a review of specialized difficulties in MAL just as profound RL ways to deal with these difficulties. Our goal is to outline a recent and active area (i.e., MDRL), as well as to propel future research to exploit the sufficient and existing writing in Multi-agent learning (MAL). We hope that the topics discussed simply will be addressed to help the new researchers. We expect that researchers with experience on either DRL or MAL could benefit from this article to understanding about MDRL and to avoid having scattered sub communities with little interaction [19],[26],[27], [34]. In the next part, we give a short-term explanation of RL.

2. LITERATURE REVIEW

REINFORCEMENT LEARNING (RL)

As mentioned, RL is a TE process, so it works by collaborating straightforwardly with environment to self-educate after some time and accomplish assigning objective and goal. RL characterizes agent or learner and define of an environment is anything outside the agent. We can show with three basic components the communication among the agent and its environment: state, action, and reward, which are displayed with the s, a, r, symbols, respectively as showed in (Fig. 2)[35]. At time-step t that shows with S_t that is based on the condition of the environment, the agent analyses S_t and performs an action a_t. The environment at that point changes S_t to S_{t+1} and gives a reward $r_{t+1}$to the agent.

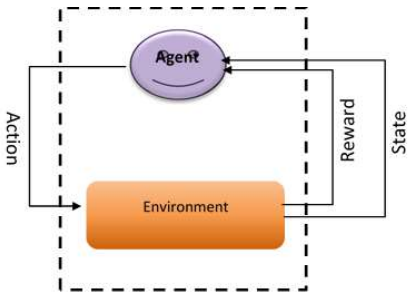


Fig. 2. Components of RL [18]

The point that we mention here is that, we can show RL problems as MDP. In fact , a MDP can be represented by a polynomial as *(S, A, T , R, γ)*, Each of these symbols represents these cases *S*: a finite set of states over time, *A* : a finite set of actions , notice that the agent interact with the environment based on set of actions, $T : S \times A \times S \rightarrow [0; 1]$ represent stochastic transition function, where $T^{s'}_{s,a}$describes the probability of arriving in state s after performing action an in state s, $R : S \times A \times S \rightarrow R$ is a scalar reward function, where $R^{s'}_{s,a}$denotes the expected reward after a state

transition, and γ is the discount factor controlling future versus immediate rewards[36]. Main fundamental idea in RL is the Markov property that is just the present state influences the following state, or in other hand, what's to come is restrictively autonomous of the past given the present state. This implies any choices made at s_t is based on s_{t-1}, not $\{s_0, s_1, \ldots, s_{t-1}\}$. Mentioned that it requires the states to be completely observable. Another concept that needs to be explained is policy. In general, the policy π means what to do? And is a mapping from states to a probability distribution over actions: $\pi: S \rightarrow p\ (A = a|S)$. If the MDP is episodic, i.e., the state is reset after each episode of length T, then the sequence of states, actions and rewards in an episode constitutes a trajectory or rollout of the policy. Every rollout of a policy accumulates rewards from the environment, resulting in the return $R=\sum_{t=0}^{T-1} \gamma^t r_{t+1}$. The goal of RL is to find an optimal policy, π^*, which achieves the maximum expected return from all states [13]:

$$\Pi^* = arg\ \max_{\pi} E[R|\pi] ,$$

Solving an MDP will yield a policy $\pi: S \rightarrow A$, an optimal policy $\pi*$ is used to maximize the expected discounted sum of rewards [36]. One of the most common techniques for solving MDPs is Bellman equation, which is based on the value iteration algorithm, which is presented in (1).

$$V^{\pi}(s) = \sum_{a \in A} \pi(s, a) \sum_{s' \in S} T(s, a, s')[R(s, a, s') + \gamma V^{\pi}(s')] \quad (1)$$

We will give a more detailed explanation of this equation, because a preliminary is needed to understand this equation.

In non-episodic ($T = \infty$) situation, $\gamma < 1$ keeps an infinite total of rewards from being aggregated. Also, strategies that depend on complete directions are never again relevant, yet those that utilization a limited set of transitions still are. A generality of MDPs are based on partially observable MDPs (POMDPs), where the agent use an observation $o_t \in \Omega$, that p $(o_{t+1} | s_{t+1}, a_t)$ is distribution of the observation and it is based on the current state and the previous action [37]. MDPs are suitable models in single agent environments to obtain optimal decisions.

2.1 Reinforcement Learning Algorithms

Model-based algorithms and Model-Free algorithms are two major classifications in RL in the following we will define each of them [35]. taxonomy of deep reinforcement algorithms shown in (Fig. 3).

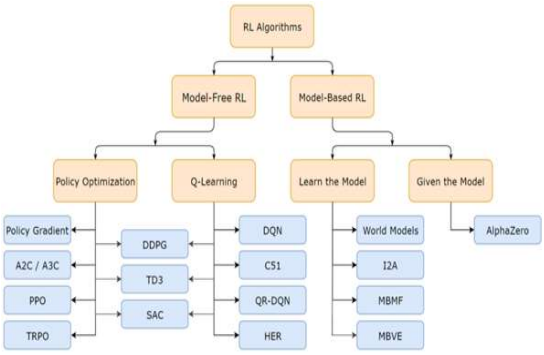


Fig. 3. Taxonomy of deep reinforcement algorithms [35]

When we want to understand how the environment works (Dynamics) from its perceptions, and then arrangement an answer utilizing that model we need to use Model-Free algorithms. Based on a model, they try to use of some technique to find optimal policy. the problem in here is that they fail in state and space that are too huge. Dynamic programming (DP) strategies are a case of model-based techniques, as they require the total knowledge of the environment [35].

Second Model –Free, i.e. This type of algorithm does not require the storage of all activities and state spaces, and the complete environment learning in these algorithms is not necessary. There are two different types of such algorithms based on the objective of the training. Policy-based and Value-based techniques [35]. As their name suggests, policy-based approaches aim at finding optimal policies, and stating that this optimal policy is either stochastic or deterministic. Policy gradients and REINFORCE algorithms have a place in this class [13]. The advantage of these algorithms can be shown in continuous action spaces and also high-dimensional problems, which makes them more convergent and more effective in these situations. Policy based techniques are basically an enhancement issue, where we find the maximum of a policy function. That is the reason we additionally use algorithms like hill climbing. Second Value- based strategies that such algorithms are similar to their name trying to find the optimal value. Q-Learning algorithms in this case will be used to find the optimal value [13].

Actor-Critic strategies is decussating between methods based on policy and value, where the objective of this method is based on optimize both Value and policy [17]. Again, we note that the goal of RL is to find optimal policies, but to find this optimal policy we need to know what to do and which action should take when we are in a certain situation and as a result of this action, we get the maximum reward. The Value function is one approach to find the optimal policy. Indeed, two value functions used today. V(s) that shows state value function and Q (s, a) that is based on the action value function. V(s) is the expected and normal return accomplished when acting from a state as indicated by the Policy (π) and Q (s, a) that is the expected return given the state and the action. Their differences are that, the first value is the estimation of a specific state. Q (s, a) is the estimation of that state in addition to the estimations of all the potential activities from that state. When we have Q (s, a), based on action with highest value, we select and perfume the action.

The Q-value can be learned by TE. So, we find the Q value, pick an activity and do it, But the evaluation of this action is based on the amount of rewards received and updating the amount of Q. It should be noted that the agent motive may initially be random, but it also explores the environment, and for each state and action the best Q-value will find by algorithm. This can be showed mathematically with use the above equation (Formula 1.) that plays a huge part in RL today's research[35].it's the time that we describe Bellman equation: The Q-value, otherwise known as the greatest future reward for a state and action, and include maximum future reward for the following state and also immediate reward. Gamma (γ) is a range between zero and one، and it's used to discount the reward as the time passes and shows that action at the beginning is more important than at the end [35]. To understand concept of this, we give an example: Suppose you ride a helicopter and its fuel cannot be enough to reach you to destination. Compared to when you spend refueling, which delays the fast reach of your destination, you prefer to refuel this time; otherwise, it may not reach the destination at all, for this reason Gamma is required.so, Q-value iteratively will be updated. In fact, this equation used to obtain the optimal policy (π^*) because it shows value of a state,

$\pi^* = argmax_\pi V^\pi(s)$,i.e., the one that maximizes that value function, and the optimal value function $V^*(s)$ [35] as described in (2).

$$V^*(s). = max_\pi V^\pi(s) \quad \forall s \in S. \quad (2)$$

states, activities, transition and also rewards should be clearly represented with value iteration and it is difficult in many situations. That's why, RL algorithms interacting with the environment in discrete time-steps and often learn from experience. The fundamental idea to comprehend here is that the Bellman condition relates states with one another and in this manner, it relates Action value functions. Iteration in the environment will help us to find and compute the optimal values, and therefore based on optimal values we can calculate the optimal policy as well. Q-value can be showed with a matrix that states are rows and actions as columns. This matrix is randomly assigned; updating this matrix is based on the calculated Q-value. In such a way that the agent performs interaction with the environment and calculates the amount of reward received from the environment for each of the actions that it performs. This is concept of Q-Learning.in the following we will explain Actor-Critics and improving by A2C and A3C.

The purpose of Actor-Critics is to take all the strengths of the value and policy-based methods and also to eliminate their weaknesses. This method can be divided into two parts, one of which is to select the action according to the state in which it is located and the next part to produce the action Q-values. If we want to specify the role of Actor and Critics in this method, it will be that the actor of receives the state as input and the output will be the best action in fact it can use a neural network like fully connected neural network or a convolutional as function approximator and it essentially controls how the agent behaves by learning the optimal policy (policy-based) . The role of critic is computing the value function for evaluates the action, in fact it is also a function approximator, input of critic is the action by the actor and the environment, interlock them and output the Q-value for the given pair. Those two models participate in a game where they both get better in their own role as the time passes. The result is that the overall architecture will learn to play the game more efficiently than the two methods separately (Fig. 4).

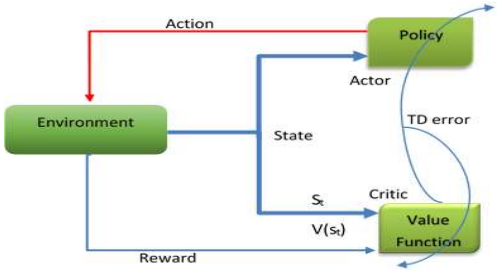


Fig. 4. Actor-Critics [35, 78]

Two networks training is separately and to find the global maximum it uses gradient ascent to update both their weights. As time passes, the actor trying for its learning and produces better actions than previous ones and the critic seeks to make better evaluations of those actions than before. It is important to notice that the update of the weights happen at each step (TD Learning) and not at the end of the episode, opposed to policy gradients [35]. Actor critics used in lots of famous 2d and 3d games, such as Doom, Super Mario, and others. two very popular improvements of Actor-critic models are based on Advantage A2C(Actor-Critic) and A3C (Asynchronous Advantage Actor-Critic).Q-values include two parts: Q(s, a)= V(s)+ A(s, a) that are based on the state Value function V(s) and the advantage value A(s, a).A(s, a) catches how better an activity is contrasted with the others at a given state, and the value function catches that it is so great to be at this state. Instead of having the critic to learn the Q-values, we make it learn the Advantage values [35]. The evaluation way of an action is based on how much better it can be. The high variance of policy networks and stabilizes the model will be reduced by advantage function [35], This is secret behind the A2C. A3C's released by DeepMind in 2016 and became important in the scientific community. The Asynchronous part is key difference from A2C. In fact, A3C is composed of several networks (agents) that work independently and each of these networks(agents) agents has its own weight, and interact with a different copy of the environment in parallel and update periodically a global network, which holds shared parameters. The asynchronous comes because updates are not happening simultaneously. With this in mind, it's clear that they can discover more areas from the state and action space in less time. After each update, the agents reset their parameters to those of the global network and continue their independent exploration and training for n steps until they update themselves again. We see that the information flows not only from the agents to the global network but also between agents as each agent resets his weights by the global network, which has the information of all the other agents [35]. Asynchrony has some problems as some agents will be involved with old version parameters and it is main drawback of asynchrony. We have an improved version of A2C with multiple agents instead of one. A2C will wait for all the agents to finish their segment and then update the global network weights and reset all the agents. The idea of combining policy and value-based method is now, in 2018, considered standard for solving reinforcement learning problems. If we want to mention the modern algorithms based on the actor-critics, we can refer to TRPO (Trust Region Policy Optimization), PPO (Proximal Policy Optimization), and DDPG (Deep Deterministic Policy Gradients).

Another case of Reinforcement Learning algorithm is the Learning Automata (LA). A Learning Automata is a Machine Learning algorithm that has been studied since the 1970s. Learning Automata can select their current actions based on past experiences in the environment. If the environment is random and the Markov Decision Process (MDP) is used, you will enter the realm of

Fig. 5. Deep Q-learning and Q-Learning [78]

an adaptive decision-making unit situated in a random environment that learns the optimal action through repeated interactions with its environment. The actions are chosen according to a specific probability distribution which is updated based on the environment response the automaton obtains by performing a particular action. LA combines quick and exact merging with low comp utational complexity and has been connected to a wide extend of modeling and control issues. In any case, the instinctive, however, the systematically tractab le concept of learning automata makes them too exceptionally appropriate as a hypothetical system for Multi-agent Reinforcement Learning (MARL). For further reading, we refer readers to [79].

2.2 Drl (Deep Reinforcement Learning)

DNNs (Deep neural networks) are utilized for demonstrate the environment dynamics (mode-based), to improve policy searches as well approximate the Value function. Research on the last one has delivered a model called Deep Q Network (Fig. 5), we can use simple Neural Networks also Convolutional, Recurrent and many else as well.

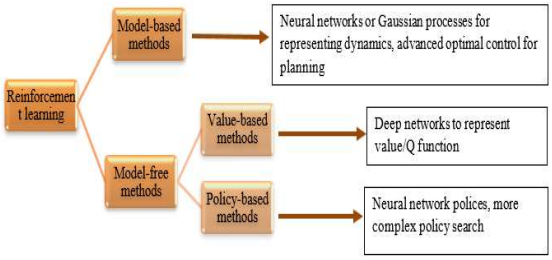
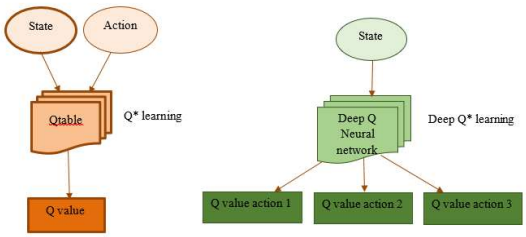


Fig. 6. types of RL models[78]

Q-learning is good. But it is not useful in situations that include big states and also unknown states because it can't derive the Q-vale of new states from the past ones. Envision a situation with

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

103 states and 103 activities for every state. It requires a table of 106 cells. Furthermore, that is a little state space appearing differently in relation to chess or Go. Imagine a scenario in which we estimated the Q-values utilizing some machine learning methods or using neural network as approximator. This scenario was taken by DeepMind by Google. Deep Q-learning use NNs (Neural Networks) as approximator for find the Q-value. Input of network is environment state and outputs are the Q-values for all actions that are possible, the Q-value with maximum amount is the action that agent should performs (Fig. 6)



Training of the agent is based on difference of value with maximum amount for next state and also current Q-value as the Bellman equation proposes. With using NNs we approximate the Q table. but there are a couple of issues that emerge. The main issue is moving Q-targets. The primary part of the TD error is the Q-Target and it is determined as the immediate reward in addition to the discounted max Q-value for the following state. When agent is training, the weights based on the TD Error is updating. But the problem is that, similar weights put on to both the predicted value and target. But there is a problem in here that is when we move the output closer to the target, the target also move. Wouldn't be incredible to keep the objective fixed as we train the network. All things considered, DeepMind did precisely that. Rather than utilizing one Neural Network, it utilizes two that named as Fixed Q-Targets. One as the fundamental Deep Q Network and a second one is Network Target to refresh only and occasionally the weights of the target. Another point that we should mention is Maximization Bias that is the tendency of DQNs to overestimate both the value functions. In fact, we can study the Maximization Bias for reasons that make Double Deep Q Network. Suppose for unknown reasons the network overestimates a Q-value for an action and that action will be picked as the go-to action for the following stage and the equivalent overestimated value will be utilized as an objective value. In other words, there is no real way to assess if the action with the maximum value is really the best activity. Hence it happens for taking care of this issue we have to utilize strategy that named, Double Deep Q Network. To report maximization bias, it is possible to use two Deep Q Networks. Responsibility of The DQN is determination of the following action also the Target network is responsible of the assessment of the target value so solve the moving target problem. Target q-value generation with decoupling the action selection will solve overestimation problem and train faster. Another important issue that needs to be mentioned is Dueling Deep Q Networks. Q-values compare to a measurement of how great an action is for a specific state this is the reason it is an action value function. The measurement is only the normal and predictable return of that activity and action from the state. Q-values can, be decayed into two parts: the state Value function V(s) and advantage value A (s, a). we simply present one more capacity: Q (s, a) = V(s)+ A (s, a) Advantage function catches how better an action is contrasted with the others at a given state, while the value function imprisonments how good it is to be at this state. The entire thought behind Dueling Q Networks depends on the portrayal of the Q function as a whole of the Value and the advantage function. Two networks are there, to get familiar with each piece of the entirety and after that we summation their outputs. The agents in now ready to assess a state without thinking about the impact of each activity from that state as mentioned, we frame RL problems as Markov Decision Processes and the goal of RL is to find the best policy, and policy can be defined as a mapping from states to actions. In other words, we want to find the action with the maximum expected reward from a given state., we achieve that in value-based methods by finding Value function and then extract the Policy. It is possible also that we find directly the Policy. This is what Policy-based methods [35]. Q-learning and Deep Q networks are great, and they are used in so many application, but Policy-based methods offer some different advantages for example they converge more easily to a local or global maximum and they don't suffer from oscillation, They have grate effective in high-dimensional or continuous spaces, and also they can learn stochastic policies (Stochastic policies give a probability distribution over actions and not a deterministic action). They used in stochastic environments, which they modelled as Partially Observable Markov Decision Processes (POMDP) where we do not know for sure the result of each action. Policy based reinforcement learning is an optimization problem, so we have a policy (π) with some parameters (θ) that outputs a probability distribution over actions. The goal is to find the best

theta because based on best theta we can find the best policy. Policy objective function J (θ) is responsible for evaluate the theta that is good or no, and also this function is varies in episodic or continuing environments accordance to (3).

$$\pi_\theta(a|s) = P[a|s]$$
$$J(\theta) = E_{\pi\theta}[\sum \gamma r] \quad (3)$$

So, the goal is to find the parameters theta (θ) that maximizes J(θ) and we have our optimal policy. There are two types of algorithms that used. Gradient free and Gradient-based algorithms. Gradient-free method does not use derivatives. For example, Hill climbing .The second family of methods is Gradient Ascent. Here's a repeat process ate the first we need to Initialize the theta and in second part next episode will be generated then Get long-term reward so after that we need to Update theta based on reward for all time steps and Repeat the process. The gradient theta should be computing analytically otherwise the whole process goes to the trash.

2.3 Using Multi-Agent In Deep Rl

Given that MASs have able to solve complex problems through collaboration between the agents, they were able to attract a lot of attention. In a MAS, agents argue with one another and associate with the environment (Fig. 7). MDP in Multi-agent Learning area is summed up to a stochastic game. n represent the number of agents, S denote states of environment that is discrete set, and set of possible actions for each agent is based on Ai, i = 1, 2, ..., n. A = A1 × A2 × .... × An represent set of joint action for all agents. Function for state transition probability is p: S × A × S → [0, 1], r: S × A × S → Rn denote the reward function. Joint policy and action used for find value function which is represented by Vπ: S ×A → Rn [3].

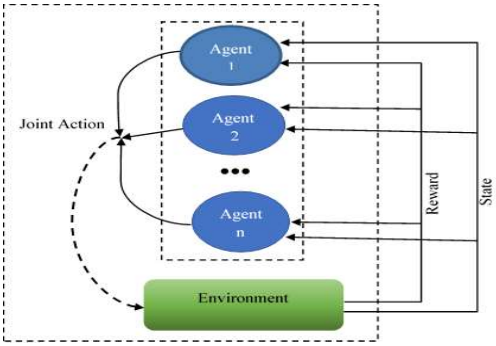


Fig. 7. Multi-Agent in Deep RL [78]

In this section we specify 4 categories take from [18], [10], [23], [27],[17] that define and characterize current works. Analysis of emergent behaviors, learning communication, learning cooperation, and agents modelling agents [17]. First works are not focus on learning algorithms and they are used to DRL algorithms for analyzing and evaluating, e.g., DQN[38],[34], [39] and others[40], [41], [42], in a Multi-agent domain. Learning communication [43], [44],[45], these works investigate a sub-territory that is pulling in consideration and that had not been investigated much in the MAL review. Learning third one is based on while learning to communicate is rising zone, encouraging collaboration in learning agents has a long history of research in MAL [23, 24], both value- based methods[46],[47], [48],[49], [50], [51],[52],[53] and policy gradients methods [45],[54],[55] used in this category. Agents modelling agents presented by Albrecht and Stone [27] and other works that are related to this topic are [56], [57],[58], that taking inspiration from DRL and [53], [59],[60],[61],[62]that are from MAL. Modelling agents is useful for modelling opponents [53],[56], [58], and [59], inferring hidden goals [57], and accounting for the learning behavior of other agents [60]. Challenges in this area can be divided into different parts such as partial observability, non-stationarity and non-stop action spaces. In this section we will briefly review each of these challenges and mention the work done in these areas. First partial observability, i.e. Conditions as partial observability is in many real-world problems, therefore, the agent does not obtain complete information about the environment. In such positions, the agents watch incomplete data about environment, and need to settle on the best action at each time. This kind of issue can be demonstrated utilizing the POMDP (Partially Observable Markov Decision Process). Various DRL models have been proposed to deal with POMDP. Hausknecht and Stone [63]proposed DRQN (Deep Recurrent Q-network) based on a LSTM (Long Short-Term Memory). The DRQN-based agents can improve policy in a strong sense in the POMDP. Not like DQN, DRQN used a RNN (Recurrent Neural Network) for finding a Q-function which is with observation o and action a, Q (o, a),[3]. In [81], DRQN is extended to DDRQN (Deep Distributed Recurrent Q-network) to evaluate in Multi-agent domain based on POMDP problems. The accomplishment of DDRQN is based on last-activity inputs, between agent weight sharing, and impairing knowledge replay. Q-function in DDRQN is based on $Q(o_t^m, oh_{t-1}^m, m, a_{t-1}^m, a_t^m; \theta_i)$ that index m shows the input of each agent [3]. Weight sharing declines learning time since it lessens the quantity of parameters to be scholarly. Gupta et al. [64] Stretched out the curriculum learning that coordinates with three classes of DRL in MAS, including actor-critic techniques policy gradient

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

and TD error. The curriculum rule is to begin figuring out how to finish basic errands first to collect information before continuing to perform muddled assignments. This is appropriate with a MAS situation where less agents at first team up before reaching out to oblige an ever-increasing number of agents to finish progressively troublesome errands. Exploratory outcomes demonstrate the imperativeness of the curriculum technique in scaling DRL calculations to complex Multi-agent issues. A framework for Multi-agent domain is demonstrate by Hong et al. [58] based on DPIQN (Deep Policy Inference Q-Network) and its upgraded based on DRPIQN (Deep Recurrent Policy Inference Q-arrange) to partial observability domain. DPIQN and DRPIQN are learned by adjusting system's regard for policy and their own Q-values at different phases of the preparation procedure. Examinations demonstrate the better presentation of both DRPIQN and DPIQN over the benchmark DQN and DRQN [63]. Integrates hysteretic learners [65] DRQNs [63], distillation [66], and CERTs (Concurrent Experience Replay Trajectories) by Omidshafiei et al. [48] Also in the context of partial observability, but extended to Multi-Task, Multi-Agent problems, Multi-Task Multi-Agent RL (MT-MARL), which are a decentralized extension of experience replay strategy proposed in [4]. The agents are not unequivocally equipped with errand personality while they figure out how to finish a lot of decentralized POMDP assignments with inadequate prizes. This strategy anyway has a disservice that can't perform in a situation with heterogeneous agents. Aside from partial observability, there are conditions that agents must arrangement with amazingly observations that are noisy, which are feebly corresponded with the genuine condition of environment. Kilinc and Montana [67] presented a strategy signified as MADDPG-M that consolidates DDPG and a correspondence medium to address these conditions. Agents should choose whether their perceptions are enlightening to impart to different agents and the correspondence policies are found out simultaneously with the fundamental strategies through experience. As of late, Foerster et al. [68] suggested BAD (Bayesian Activity Decoder) calculation for learning various agents with settings in partial observable situations. BAD depends on a factorized and rough conviction state to find ideal policies by agents [3]. Second non-stationarity i.e. Controlling various agents represents a few extra difficulties when contrasted with single agent condition, for example, the heterogeneity of agents, how to characterize reasonable aggregate objectives or the versatility to enormous amount of agents that needs structure of minimal portrayals, and all the more critically the non-stationary issue. An agent in single condition is evaluating just the result of behavior of itself, but in MA space, watches the results of its own activity as well as the conduct of different agents. Learning between the agents is intricate in light of the fact that all operators conceivably associate with one another and adapt simultaneously [3]. Non-stationarity and reshape the environment is based on associations among multiple agents. Therefore, learning among the agents let to changes in the policy of an agent, so influence the ideal policy of other agents. In the future, good policy will not remain for the reason that ,the evaluated possible rewards of an activity would be wrong. In the non-stationary condition, Q-learning connected in single agent setting isn't applied to most Multi-agent issues because the Markov property does not hold any more in this condition [26]. In this manner, agents' stability should be remaining by performing with certain recurrence based on collecting and processing of information [3]. The DQN [4] and Q-learning[69] was not intended for the non-stationary situations. Two types of DQN suggested by Castaneda [70], in particular DRUQN (Deep Repeated Update Q-Network) and DLCQN (Deep Loosely Coupled Q-Network), used to manage the non-stationarity issue in MAS [3]. The DRUQN is created dependent on the RUQL (Repeated Update Q-learning) model presented in [71], [72] that is based on DLCQN (Loosely Coupled Q-learning). The aims of DRUQN is using its observation and negative rewards for determines of independence degree in each agent [3]. Independent degree responsible is, the agent figures out how to choose whether it needs to act freely or collaborate with different agents in various conditions. Another case that was used in non-stationary conditions was Diallo et al. [73] that stretched out DQN to a Multi-agent concurrent DQN. Foerster et al. [46] presented two strategies for balancing out experience replay of DQN in a MADRL Palmer eta. [47] Showed a technique that named LDQN (lenient-DQN) in non-stationarity because of concurrent learning of different agents in MAS that is for updating policy from experience replay memory [3]. That technique is connected to the organized Multi-agent object transportation issues and its exhibition is contrasted and the HDQN (hysteretic-DQN) [48]. The test results show the predominance of LDQN against HDQN as far as assembly to ideal policies in a stochastic reward condition [3]. WDDQN (weighted twofold profound Q-arrange) in [49] proposed to manage non-stationary in MAS. Trials demonstrate the better execution of WDDQN against DDQN in two Multi-agent situations with stochastic rewards and enormous state space [3]. And third one is based on

non-stop action spaces, i.e., Furthermost DRL must be connected to Non-stop action spaces [74]. For instance, DQN is restricted distinctly to issues low-

dimensional with separate activity domains [4], it can deal with high-dimensional spaces, DQN intends to discover activity that has most extreme action -value, and along these lines requires an iterative enhancement process at each progression in the nonstop activity state. Discretizing the activity space is a conceivable answer for adjust DRL techniques to non-stop areas. This makes numerous issues, remarkably is the scourge of dimensionality: the exponential increment of activity numbers against the quantity of degrees of opportunity. TRPO (Trust Region Policy Optimization) technique offered by Schulman et al. [75] that can be stretched to non-stop actions and states, for improving stochastic control policies in the area of robotic loco motion and game playing based on image based. An off-Policy method, to be DDPG (Deep Deterministic Policy Gradient), which uses the actor-critic design [76] to deal with the non-stop activity spaces presented by Lillicrap et al. [74]. In view of the DPG (Deterministic Policy Gradient), DDPG deterministically maps states to explicit activities utilizing a parameterized actor function while keeping DQN learning on the critic side. DDPG to RDPG (Recurrent DPG) stretched out to deal with issues with non-stop activity spaces based on partial observability by Heess et al.[77], where the genuine state isn't accessible to the agents when deciding. As of late, PS-TRPO technique for MAL presented by Gupta et al. [64], this technique depends on the establishment of TRPO so it can manage non-stop activity spaces successfully. Table 1 summarizes of important papers mentioned in the MADRL section.as mentioned, in this section we specify 4 categories take from [18], [10], [23], [27],[17] that define and characterize current works. Analysis of emergent behaviors, learning communication, learning cooperation, and agents modelling agents [17].

Table 1. Summarizes of important papers mentioned in the MADRL section.

MADRL Emergent behaviors		
Paper	Brief Description	
Raghu et al. [39]	Attacker- Defender game using PPO, DQN and A3C	
Tampuu et al. [38]	Play Pong with Multi Agents DQN	
Lazaridou et al. [40]	Communication language with NN with using agents.	
Bansal et al. [42]	MuJoCo using competitive PPO agents	
Learning Communication Methods		
Structure	Method	Brief Description
DRQN	DIAL [43]	Execution is based on communication actions and learning process is based on gradient sharing
Multi-Layer NN	CommNet [44]	Communication on single network done with continuous vector channel.
Bidirectional RNN	BiCNet[45]	AC method is used for communication in latent space
Learning cooperation		
Structure	Method	Brief Description
IQN	Fingerprints [46]	Conditioning the value function on fingerprint with MADRL, that remove uncertainty age of sampled data
DRQN	HYSTERETIC – drqn[48]	Using 2 rates of learning for take cooperation
DDPG	MADDPG [55]	Critic is completed with other agent's information based on AC method
Agents Modeling Agents		
Method	Brief Description	
DRON	DQN method used in network to infer the opponent manner	
DPIQN, DPIRQN	Raw observation used for learning policy features, which indicate high level opponent behaviors through auxiliary tasks	
DCH	Policies can overfit to opponents, best compute proximate better answer to policy mixture	

Since, the mentioned methods have many applications in various subjects such as medical images, robotics, games, economic, agriculture and so on, only papers that have used these methods have been mentioned. Their results can be checked by referring to any article. Since they have been used in various applications.
Since DRL and MADRL methods are used in different cases and sciences, as a result, each of the reviewed papers has different datasets to conduct their research. Therefore, readers are advised to refer to the papers for information about datasets.

3. DISCUSSION AND FUTURE WORK

As mentioned in the previous sections, a lot of work has been done based on DRL in combination with Multi-Agents. One of the most practical aspects of MADRL is the use of multi-agents in medical imaging. very little work has been done on the processing of medical images based on RL. For example, by [36], in the heart and brain, based on DRL method, Landmark Detection is done. Landmark detection is the most important steps in processing of medical images. In Many computer vision applications Multi Object Tracking has been a key research subject. Considering that manual handling of these processes is accompanied by errors and loss of time the automation of them is a pressing need for today's society. Although medical imaging companies have developed software, but this type of software is not fully automated and very expensive or is limited to use with images that have been acquired on specific manufactures' machines.

we are trying to automate the Multi Landmark detection and Multi Object Tracking of Left Heart (LH) and Right Heart (RH) in medical images using MADRL. According to our research, the best method used by Multi-agents to determine Multi landmark detection is Duel DQN, as mentioned in [103], Duel DQN can achieve more robust estimates of state value by decoupling it from specific actions and showed better results than the previous baselines of DQN and DDQN on several Atari games. Broadly, duel DQN and DDQN introduced vast improvements in performance compared to DQN, yet it does not necessarily result in better performance in all environments. For object detection we are trying to use YoLo V3 [104], which is a state-of-the-art and real time object detection system. After detection of each object we are trying to use CNN and DQN in MADRL part, to find the movement function of the heart muscles.

There is a lot of work to be done in the field of medical imaging, such as automatic segmentation of images using deep reinforcement learning. Each of these issues will be very interesting topics for research

4. CONCLUSION

DRL has had many successes in various fields [11, 12, and 18], The use of Multi-agents in this field can be considered as another step to success in DRL. As it clear, Multi-Agent Learning is much harder than Single –agent learning due to high-dimensionality, non-stationarity [1],[2],[8],[43]. In this article, we review new works on Multi-agent Learning, as well as an explanation of previous works and other approaches in this field.

REFERENCES

[1] H. Hoag, "Animal intelligence," *Nature*, vol. 441, no. 7092, pp. 544–545, 2006, doi: 10.1038/nj7092-544a.
[2] I. V. Bajić and J. W. Woods, "Maximum minimal distance partitioning of the $\mathbb{Z}2$ lattice," *IEEE Trans. Inf. Theory*, vol. 49, no. 4, pp. 981–992, 2003, doi: 10.1109/TIT.2003.809572.
[3] T. T. Nguyen, N. D. Nguyen, and S. Nahavandi, "Deep Reinforcement Learning for Multi-Agent Systems: A Review of Challenges, Solutions and Applications," no. 1992, pp. 1–27, 1997, doi: arXiv:1812.11794v1.
[4] Y. Zhan, H. B. Ammar, and M. E. Taylor, "Human-level control through deep reinforcement learning," *IJCAI Int. Jt. Conf. Artif. Intell.*, vol. 2016-Janua, no. 7540, pp. 2315–2321, 2016, doi: 10.1038/nature14236.
[5] S. Mahadevan and J. Connell, "Automatic programming of behavior-based robots using reinforcement learning," *Artif. Intell.*, vol. 55, no. 2–3, pp. 311–365, 1992, doi: 10.1016/0004-3702(92)90058-6.
[6] Y. Mohammad and T. Nishida, "Learning from demonstration," *Adv. Inf. Knowl. Process.*, no. 9783319252308, pp. 293–317, 2015, doi: 10.1007/978-3-319-25232-2_13.
[7] H. Benbrahim, "learning," 1996.
[8] M. Riedmiller, T. Gabel, R. Hafner, and S. Lange, "Reinforcement learning for robot soccer," *Auton. Robots*, vol. 27, no. 1, pp. 55–73, 2009, doi: 10.1007/s10514-009-9120-4.
[9] K. Mülling, J. Kober, O. Kroemer, and J. Peters, "Learning to select and generalize striking movements in robot table tennis

Cited by me DMP reactive_motion_gen...," *Int. J. Robot. ...*, pp. 1–24, 2013.

[10] L. Buşoniu, R. Babuška, and B. De Schutter, "Lucian Busoniu, Robert Babuska, and Bart De Schutter. A comprehensive survey of multiagent reinforcement learning. IEEE Transactions on Systems Man and Cybernetics Part C Applications and Reviews, 38(2):156, 2008.," *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.*, vol. 38, no. 2, pp. 156–172, 2008.

[11] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, "The arcade learning environment: An evaluation platform for general agents," *IJCAI Int. Jt. Conf. Artif. Intell.*, vol. 2015-Janua, no. Ijcai, pp. 4148–4152, 2015.

[12] L. Deng and D. Yu, "Deep learning: Methods and applications," *Found. Trends Signal Process.*, vol. 7, no. 3–4, pp. 197–387, 2013, doi: 10.1561/2000000039.

[13] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 26–38, 2017, doi: 10.1109/MSP.2017.2743240.

[14] J. Ker, L. Wang, J. Rao, and T. Lim, "Deep Learning Applications in Medical Image Analysis," *IEEE Access*, vol. 6, pp. 9375–9379, 2017, doi: 10.1109/ACCESS.2017.2788044.

[15] G. Litjens *et al.*, "A survey on deep learning in medical image analysis," *Med. Image Anal.*, vol. 42, no. December 2012, pp. 60–88, 2017, doi: 10.1016/j.media.2017.07.005.

[16] A. S. Lundervold and A. Lundervold, "An overview of deep learning in medical imaging focusing on MRI," *Z. Med. Phys.*, vol. 29, no. 2, pp. 102–127, 2019, doi: 10.1016/j.zemedi.2018.11.002.

[17] P. Hernandez-Leal, B. Kartal, and M. E. Taylor, "Is multiagent deep reinforcement learning the answer or the question A brief survey.pdf," *Auton. Agent. Multi. Agent. Syst.*, vol. 33, no. 6, pp. 750–797, 2019, doi: 10.1007/s10458-019-09421-1.

[18] P. Stone and M. Veloso, "Multiagent Systems : A Survey from a Machine Learning Perspective 1 Introduction 2 Multiagent Systems," pp. 1–57, 1997.

[19] Y. Shoham, R. Powers, and T. Grenager, "If multi-agent learning is the answer , what is the question ?," pp. 1–21, 2006.

[20] E. Alonso, M. Inverno, D. Kudenko, M. Luck, and J. Noble, "Learning in Multi-Agent Systems," no. June, 2001.

[21] K. Tuyls and G. Weiss, "Multiagent learning: Basics, challenges, and prospects," *AI Mag.*, vol. 33, no. 3, pp. 41–52, 2012, doi: 10.1609/aimag.v33i3.2426.

[22] M. Wiering and van O. Martijn, *Reinforcement Learning: State-of-the-Art*, no. December 2014. 2013.

[23] M. Kestin, I. L. Rouse, R. A. Correll, and P. J. Nestel, "Cardiovascular disease risk factors in free-living men: Comparison of two prudent diets, one based on lactoovovegetarianism and the other allowing lean meat," *Am. J. Clin. Nutr.*, vol. 50, no. 2, pp. 280–287, 1989, doi: 10.1093/ajcn/50.2.280.

[24] L. Matignon, G. J. Laurent, and N. Le Fort-Piat, "Independent reinforcement learners in cooperative Markov games: A survey regarding coordination problems," *Knowl. Eng. Rev.*, vol. 27, no. 1, pp. 1–31, 2012, doi: 10.1017/S0269888912000057.

[25] D. Bloembergen, K. Tuyls, D. Hennes, and M. Kaisers, "Evolutionary dynamics of multi-agent learning: A survey," *J. Artif. Intell. Res.*, vol. 53, no. August, pp. 659–697, 2015, doi: 10.1613/jair.4818.

[26] P. Hernandez-Leal, M. Kaisers, T. Baarslag, and E. M. de Cote, "A Survey of Learning in Multiagent Environments: Dealing with Non-Stationarity," pp. 1–64, 2017.

[27] S. V Albrecht and P. Stone, "Autonomous Agents Modelling Other Agents :," no. September 2017, 2018.

[28] S. Babbar, "Mastering the game of Go with deep neural networks and tree search - AlphaGoNaturePaper.pdf," no. October, pp. 2–5, 2017.

[29] D. Silver, J. Schrittwieser, K. Simonyan, I. A.- Nature, and U. 2017, "Mastering the game of Go without human knowledge," *Nature*, vol. 550, no. 7676, p. 354, 2016.

[30] M. Moravčík *et al.*, "DeepStack: Expert-level artificial intelligence in heads-up no-limit poker," *Science (80-.).*, vol. 356, no. 6337, pp. 508–513, 2017, doi: 10.1126/science.aam6960.

[31] F. L. Da Silva, M. E. Taylor, and A. H. R. Costa, "Autonomously reusing knowledge in multiagent reinforcement learning,"

IJCAI Int. Jt. Conf. Artif. Intell., vol. 2018-July, pp. 5487–5493, 2018, doi: 10.24963/ijcai.2018/774.

[32] A. K. Agogino, M. Field, and M. Field, "Unifying Temporal and Structural Credit Assignment Problems."

[33] E. Wei and S. Luke, "Lenient Learning in Independent-Learner Stochastic Cooperative Games," vol. 17, pp. 1–42, 2016.

[34] A. Darwiche, "Human-level intelligence or animal-like abilities?," *Commun. ACM*, vol. 61, no. 10, pp. 56–67, 2018, doi: 10.1145/3271625.

[35] R. Sutton and A. Barto, "Reinforcment Learning," *ACS Symp. Ser.*, vol. 674, 1997, doi: 10.1016/S1364-6613(99)01331-5.

[36] A. Alansary *et al.*, "Evaluating reinforcement learning agents for anatomical landmark detection," *Med. Image Anal.*, vol. 53, pp. 156–164, 2019, doi: 10.1016/j.media.2019.02.007.

[37] L. Pack, M. L. Littman, and A. R. Cassandra, "KaelblingLittmanCassandra1998," *Artif. Intell.*, vol. 101, pp. 1–36, 1998.

[38] A. Tampuu *et al.*, "Multiagent cooperation and competition with deep reinforcement learning," *PLoS One*, vol. 12, no. 4, pp. 1–12, 2017, doi: 10.1371/journal.pone.0172395.

[39] M. Raghu, A. Irpan, J. Andreas, R. Kleinberg, Q. Le, and J. Kleinberg, "Can Deep Reinforcement Learning Solve Erdos-Selfridge-Spencer Games ?," 2018.

[40] A. Lazaridou, A. Peysakhovich, and M. Baroni, "Multi-agent cooperation and the emergence of (natural) language," *5th Int. Conf. Learn. Represent. ICLR 2017 - Conf. Track Proc.*, pp. 1–11, 2017.

[41] I. Mordatch and P. Abbeel, "Emergence of grounded compositional language in multi-agent populations," in *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*, 2018, pp. 1495–1502.

[42] T. Bansal, J. Pachocki, S. Sidor, I. Sutskever, and I. Mordatch, "Emergent Complexity via Multi-Agent Competition," vol. 2, pp. 1–12, 2017.

[43] J. N. Foerster, Y. M. Assael, N. de Freitas, and S. Whiteson, "Learning to Communicate to Solve Riddles with Deep Distributed Recurrent Q-Networks," 2016, doi: 10.7551/ecal.

[44] S. Sukhbaatar, A. Szlam, and R. Fergus, "Learning Multiagent Communication with Backpropagation," no. Nips, 2016, doi: 10.1016/j.envsci.2011.10.005.

[45] P. Peng *et al.*, "Multiagent Bidirectionally-Coordinated Nets: Emergence of Human-level Coordination in Learning to Play StarCraft Combat Games," 2017, doi: 10.1007/11575726_13.

[46] J. Foerster, N. Nardelli, G. Farquhar, and T. Afouras, "Stabilising Experience Replay for Deep Multi-Agent Reinforcement Learning," 2017.

[47] G. Palmer and K. Tuyls, "Lenient Multi-Agent Deep Reinforcement Learning," no. July, 2018.

[48] S. Omidshafiei, J. Pazis, C. Amato, J. P. How, and J. Vian, "Deep Decentralized Multi-task Multi-Agent Reinforcement Learning under Partial Observability," 2017.

[49] Y. Zheng, J. Hao, and Z. Zhang, "Weighted Double Deep Multiagent Reinforcement Learning in Stochastic Cooperative Environments," 2018.

[50] M. Jaderberg *et al.*, "Human-level performance in first-person multiplayer games with population-based deep reinforcement learning."

[51] P. Sunehag, G. Lever, N. Sonnerat, and M. Jaderberg, "Value-Decomposition Networks For Cooperative Multi-Agent Learning," 2012.

[52] T. Rashid, M. Samvelyan, and C. Schroeder, "QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning," 2018.

[53] M. Lanctot, D. Silver, and K. Tuyls, "A Unified Game-Theoretic Approach to Multiagent Reinforcement Learning," no. Nips, 2017.

[54] J. N. Foerster, G. Farquhar, T. Afouras, N. Nardelli, S. Whiteson, and U. Kingdom, "Counterfactual Multi-Agent Policy Gradients," 2007.

[55] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *Adv. Neural Inf. Process. Syst.*, vol. 2017-Decem, pp. 6380–6391, 2017.

[56] J. Boyd-graber, K. Kwok, and H. Daum, "Opponent Modeling in Deep Reinforcement Learning," vol. 48, 2016.

[57] R. Raileanu, E. Denton, A. Szlam, and R.

Fergus, "Modeling Others using Oneself in Multi-Agent Reinforcement Learning."

[58] Z. Hong, S. Su, T. Shann, Y. Chang, and C. Lee, "A Deep Policy Inference Q-Network for Multi-Agent Systems."

[59] J. Heinrich and D. Silver, "Deep Reinforcement Learning from Self-Play in Imperfect-Information Games," 2016.

[60] J. Foerster, R. Y. Chen, M. Al-shedivat, I. Mordatch, P. Abbeel, and U. C. Berkeley, "Learning with Opponent-Learning Awareness."

[61] N. C. Rabinowitz, F. Perbet, H. F. Song, C. Zhang, S. M. A. Eslami, and M. Botvinick, "Machine Theory of Mind," 1992.

[62] T. Yang, J. Hao, Z. Meng, C. Zhang, Y. Zheng, and Z. Zheng, "Towards efficient detection and optimal response against sophisticated opponents," *IJCAI Int. Jt. Conf. Artif. Intell.*, vol. 2019-Augus, no. July, pp. 623–629, 2019, doi: 10.24963/ijcai.2019/88.

[63] M. Hausknecht and P. Stone, "Deep Recurrent Q-Learning for Partially Observable MDPs," 1997.

[64] J. K. Gupta, M. Egorov, and M. Kochenderfer, "Cooperative Multi-Agent Control Using Deep Reinforcement Learning," *Adapt. Learn. Agents 2017*, 2017, doi: 10.1007/978-3-319-71682-4.

[65] L. Matignon, G. J. Laurent, and N. Le Fort-Piat, "Hysteretic Q-Learning : An algorithm for decentralized reinforcement learning in cooperative multi-agent teams," *IEEE Int. Conf. Intell. Robot. Syst.*, pp. 64–69, 2007, doi: 10.1109/IROS.2007.4399095.

[66] A. A. Rusu *et al.*, "Policy distillation," *4th Int. Conf. Learn. Represent. ICLR 2016 - Conf. Track Proc.*, pp. 1–13, 2016.

[67] O. Kilinc and G. Montana, "Multi-agent Deep Reinforcement Learning with Extremely Noisy Observations," 2018.

[68] J. N. Foerster *et al.*, "Bayesian action decoder for deep multi-agent reinforcement learning," *36th Int. Conf. Mach. Learn. ICML 2019*, vol. 2019-June, pp. 3428–3442, 2019.

[69] M. Tan, "Multi-Agent Reinforcement Learning: Independent vs. Cooperative Agents," *Mach. Learn. Proc. 1993*, pp. 330–337, 1993, doi: 10.1016/B978-1-55860-307-3.50049-6.

[70] A. O. Castañeda, "Thesis: Deep Reinforcement Learning Variants of Multi-Agent Learning Algorithms," 2016.

[71] S. Abdallah and M. Kaisers, "Addressing the policy-bias of Q-learning by repeating updates," *12th Int. Conf. Auton. Agents Multiagent Syst. 2013, AAMAS 2013*, vol. 2, pp. 1045–1051, 2013.

[72] S. Abdallah and M. Kaisers, "Addressing environment non-stationarity by repeating Q-learning updates," *J. Mach. Learn. Res.*, vol. 17, pp. 1–31, 2016.

[73] E. A. O. Diallo, A. Sugiyama, and T. Sugawara, "Learning to coordinate with deep reinforcement learning in doubles pong game," *Proc. - 16th IEEE Int. Conf. Mach. Learn. Appl. ICMLA 2017*, vol. 2017-Decem, pp. 14–19, 2017, doi: 10.1109/ICMLA.2017.0-184.

[74] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," *4th Int. Conf. Learn. Represent. ICLR 2016 - Conf. Track Proc.*, 2016.

[75] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel, "Trust Region Policy Optimization," 2015, doi: 10.1063/1.4927398.

[76] V. Mnih *et al.*, "Asynchronous Methods for Deep Reinforcement Learning," 2016, doi: 10.1177/0956797613514093.

[77] N. Heess, J. J. Hunt, T. P. Lillicrap, and D. Silver, "Memory-based control with recurrent neural networks," pp. 1–11, 2015.

[78] V. François-lavet *et al.*, "An Introduction to Deep Reinforcement Le arning (arXiv:1811.12560v1 [cs.LG])http://arxiv.org/abs/1811.12560 ," *Found. trends Mach . Learn.*, vol. II, no. 3–4, pp. 1–140, 2018, doi: 10.1561/2200000071.Vincent.

[79] A. Nowé, K. Verbeeck, and M. Peeters, "Learning automata as a basis for multi agent reinforcement learning," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 3898 LNAI, pp. 71–85, 2006, doi: 10.1007/11691839_3.