Intelligent and Converged Networks ISSN 2708-6240
2023, 4(1): 50−75 DOI: 10.23919/ICN.2023.0005

Towards a multi-agent reinforcement learning approach for joint sensing and sharing in cognitive radio networks

Kagiso Rapetswa and Ling Cheng*

Abstract: The adoption of the Fifth Generation (5G) and beyond 5G networks is driving the demand for learning approaches that enable users to co-exist harmoniously in a multi-user distributed environment. Although resource-constrained, the Cognitive Radio (CR) has been identified as a key enabler of distributed 5G and beyond networks due to its cognitive abilities and ability to access idle spectrum opportunistically. Reinforcement learning is well suited to meet the demand for learning in 5G and beyond 5G networks because it does not require the learning agent to have prior information about the environment in which it operates. Intuitively, CRs should be enabled to implement reinforcement learning to efficiently gain opportunistic access to spectrum and co-exist with each other. However, the application of reinforcement learning is straightforward in a single-agent environment and complex and resource intensive in a multi-agent and multi-objective learning environment. In this paper, (1) we present a brief history and overview of reinforcement learning and its limitations; (2) we provide a review of recent multi-agent learning methods proposed and multi-agent learning algorithms applied in Cognitive Radio (CR) networks; and (3) we further present a novel framework for multi-CR reinforcement learning and conclude with a synopsis of future research directions and recommendations.

Key words: cognitive radio; multi-agent reinforcement learning; deep reinforcement learning; mean field reinforcement learning; organic computing

1 Introduction

The Cognitive Radio (CR) transcends Software Defined Radio (SDR) because it has cognitive abilities that enable it to observe, reason, and learn from its interactions with its operating environment[1]. Through these abilities, the CR can conduct various cognitive tasks. The CR's cognitive tasks fall under three broad categories: environmental awareness, reconfigurability, and learning. Each task is conducted using spectrum sensing, sharing, and learning techniques. The CR can identify and leverage idle spectrum through the right combination of sensing and sharing techniques. Gaining opportunistic access to idle spectrum enables unlicensed users to bypass some of the spectrum access barriers and utilise spectrum when the user to whom the spectrum has been licensed is not using it and without causing any harm to the licensed user's operations. Overlaying learning techniques over sensing and sharing techniques improves the efficiency of the CR, the quality of decisions made by the CR, and the level of protection given to the Primary User's (PU's) operations[2].

As depicted in Fig. 1, adapted from Ref. [3], the CR is designed to execute its cognitive tasks by following the cognitive cycle within its cognitive engine. The learning capability can underpin all the tasks in the cognitive cycle to unearth the benefits of learning in the CR's operations. The learning capability is designed to develop as the CR builds its understanding of its operating environment through observing, analysing, and reasoning of its experiences[2].

• Kagiso Rapetswa and Ling Cheng are with the School of Electrical and Information Engineering, University of the Witwatersrand, Johannesburg 0001, South Africa. E-mail: 060575f@students.wits.ac.za; Ling.Cheng@wits.ac.za.
* To whom correspondence should be addressed.
Manuscript received: 2023-02-09; accepted: 2023-03-31

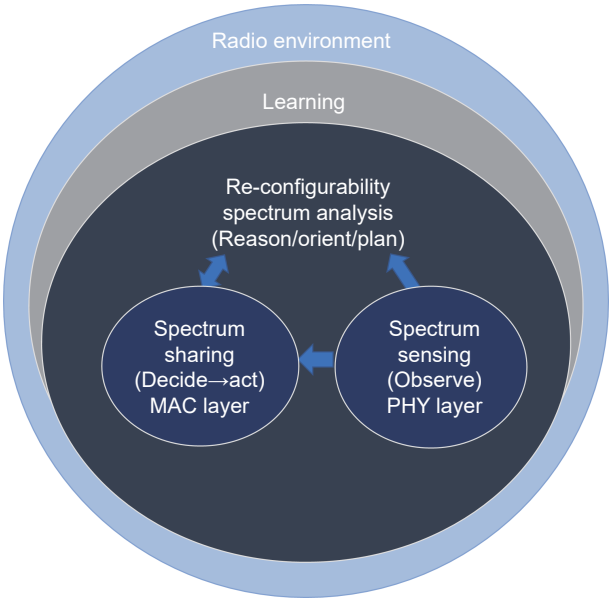

Fig. 1 Cognitive cycle underpinned by cognitive tasks.

Learning in CRs is enabled through machine learning algorithms adapted for CR environments and collectively referred to as CR learning algorithms. It is envisioned that CRs will be deployed in distributed Fifth Generation (5G) and beyond 5G networks, without any prior information about their operating environment. The most suitable CR learning algorithm for such applications is reinforcement learning as it can be used to enable the CR to intelligently optimise its sensing and sharing parameters to enable efficient opportunistic access to the spectrum, through trial and error.

Reinforcement learning is effective in single-agent and single-objective learning scenarios. However, multi-agent environments have proven to be computationally complex and resource intensive. Moreover, the higher the number of learning agents in the network, the more time it takes for the algorithm to converge and the higher the risk of the algorithm not converging to a stable point. The risk of non-convergence emanates from having multiple learning agents learning simultaneously. During multi-agent learning, each learning agent classifies the state it encounters as good or bad depending on the experience it had while on the state. An example of a state in a CR network is the channel and the CRs on that particular channel. A CR will classify the state as bad if it shared the channel with many other CRs or if interference levels or noise levels were high. This means that the learning algorithm can have more states than the number of channels. The more states there are the longer the convergence time of the algorithm, the more each CR wants to explore new states, and the longer it takes to reach convergence. These challenges are further exacerbated in multi-objective and multi-CR reinforcement learning algorithms, even more so because the CR is battery-powered and has limited computational processing power and memory to store historical experiences[4]. However, many advancements have been made to limit the negative effects of these limitations and unearth new opportunities for non-cooperative collaboration amongst learning agents in a distributed CR environment. Additional detail describing these opportunities is provided in the rest of the paper.

This paper elaborates on how the field of Multi-Agent Reinforcement Learning (MARL) has taken shape and found applications in CR networks over the last twenty years. Section 2 provides background on the CR and the field of reinforcement learning in distributed networks. It further describes the importance of multi-agent learning and how game theory has contributed to the advancement of this field. Section 3 closely examines recent developments in the field of multi-agent learning with an added focus on deep reinforcement learning developments and the gaps in the existing approaches. In Section 4, a novel approach to multi-CR and multi-objective reinforcement learning in a distributed CR network is presented. Finally, a selected set of CR MARL open research questions is discussed in Section 5. Section 6 provides a summary of the conclusions drawn.

2 Background

Mitola[5] developed the concept of the CR and defined the CR as a radio that includes forms of machine learning and employs model-based reasoning to achieve a stated objective in radio-related domains. The CR observes its environment to build its understanding of the environment and decides on appropriate action. When considering radio spectrum efficiency, the CR typically executes the below activities[3].

(1) **Sensing:** The CR applies spectrum sensing techniques to observe or identify channels used by the Primary User (PU) and vacant channels that can be accessed.

(2) **Decision:** After sensing has been concluded, the CR considers its observations, then plans and decides its next course of action.

(3) **Sharing:** The CR implements policy rules to reconfigure its parameters to transmit over the selected channel. If another CR occupies the selected channel, the incoming CR will reconfigure itself to ensure both CRs can transmit successfully over the channel.

(4) **Mobility:** If the CR observes PU activity on the channel it was transmitting on, the CR vacates the spectrum to not cause interference with the PU.

The sensing activity is usually the first activity the CR conducts. Thereafter, the CR may, using its learning and reasoning capabilities, elect to conduct the mobility, decision, or sharing activities. It is, however, the best practice for the CR to conduct the decision activity after the sensing activity so it can formally decide on the next set of actions to take. The relationship between these four activities is depicted in Fig. 2.

In this context, the sensing activity is executed as part of the awareness task, while the sharing activity is executed as part of the reconfigurability task. These activities are also referred to as resource allocation and management tasks[4].

The awareness task entails the implementation of a suitable sensing technique such as energy detection. The reconfigurability task involves enabling the CR to adjust a wide range of operating parameters —often simultaneously— based on radio communication models. These parameters include sensing, transmission power

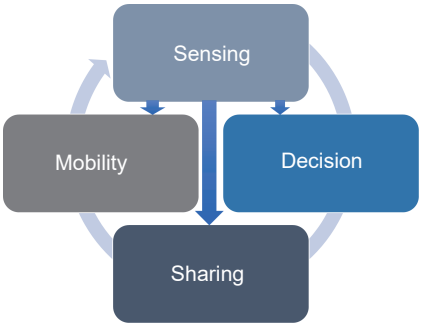

Fig. 2 Radio spectrum efficiency related cognitive activities.

channel, frame size, modulation, interference control, and routing parameters. Failure by the CR to conclude these tasks timeously can result in missed spectrum opportunities and thus low spectrum efficiency. The CR is designed to be an intelligent device that can avoid these failures by applying suitable CR learning algorithms to either or both the awareness and the reconfigurability tasks[6]. In this way, the CR will learn from its past experiences, improve its decisions, and minimise the time spent on executing the sensing and reconfigurability tasks.

CR learning algorithms are broadly categorised into (1) classification and (2) decision-making[7]. This categorisation is depicted in Fig. 3 and adapted from Ref. [7]. The algorithms categorised as classification algorithms are used to enable agents to classify data received or observed. The algorithms categorised as decision-making algorithms allow the CR to learn similarly to how a human would learn, that is, by continuously making decisions on how to act in an operating environment and tagging the experience as good or bad. Reinforcement Learning (RL) is the most prevalent class of algorithms developed to mimic human learning behaviour in new environments. The fundamental principle of RL is learning through trial and error and formulated either as a model-based or model-free RL algorithm. The CR serves as the learning agent. The model-based algorithms enable the CR to learn the underlying model or apply a given model to arrive at a learning outcome. The model-free RL algorithms allow the CR to decide on an action to take by using a value-based function or by developing or implementing a policy related to the CR's experiences.

There are times when the RL approaches were categorised as unsupervised learning approaches because of their ability to detect patterns in the environment observed without any prior information (i.e., learn the underlying model). However, RL approaches are inefficient in data categorisation because they were not designed for this purpose, unlike data categorisation approaches such as the K-means and the association rules approach.

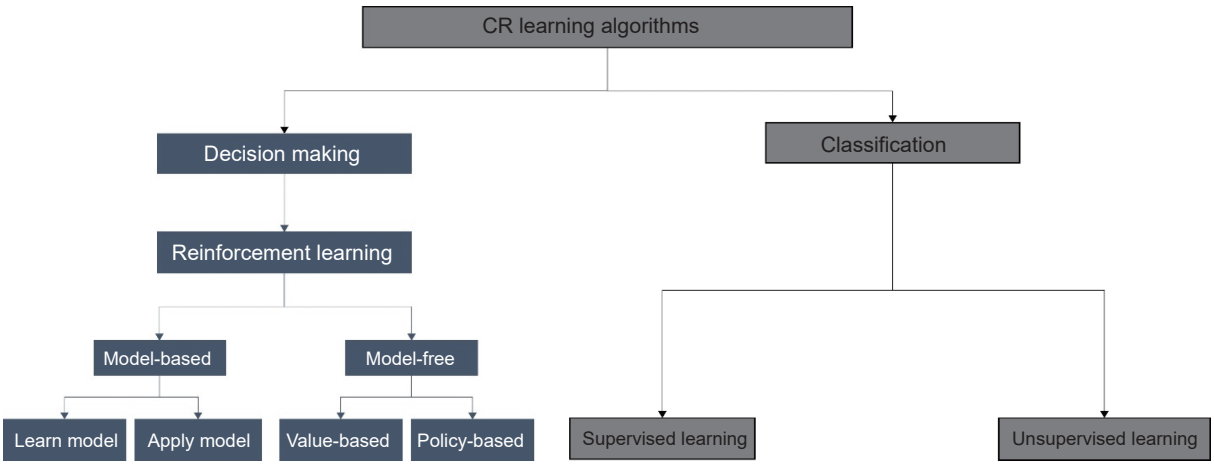

Fig. 3 CR learning categories.

RL approaches have a rich and long history stemming from Richard Bellman, Richard Sutton, Chris Watkins, Michael Littman, Marvin Minsky, Rob Howard, Paul Werbos, Andrew Barto, and other researchers. Some notable contributions in the field of RL include the Bellman equation, the temporal difference learning algorithm, the Q-learning algorithm, and the minimax-Q learning equation. Richard Bellman developed the Bellman equation. Richard Sutton developed the temporal difference learning algorithm, Chris Watkins developed the Q-learning algorithm, and Andrew Barto contributed to the development of the actor-critic algorithm. Michael Littman contributed the minimax-Q learning equation, which is the first modification of the single-agent Q-learning equation[8]. These contributions have been instrumental in positioning RL as a critical enabler of artificial intelligence, particularly in 5G networks. The Q-learning approach remains the most widely used RL approach[9]. However, Multi-Agent Reinforcement Learning (MARL) approaches require further development to fully enable the realisation of the potential that lies in Artificial Intelligence (AI) systems.

The model-based RL algorithms model the environment as a Markov Decision Process (MDP), Partially Observable MDP (POMDP), or a non-MDP with a specified state-and-action space and a reward function[10]. The model-free category is further subdivided into the policy search algorithms and the value-based algorithms. The policy search approach advocates for the search of the optimal policy over a period of time, while the value-based approach aims to optimise a value function. The collective objective of all RL approaches is for the learning agent to find a sequence of actions that will maximise its long-term reward. This objective is best met by solving Bellman's equation, which is given by Eq. (1). Bellman's equation defines a set of equations that express the utility U of each state s in the system based on the utility of the next state s' in the system, assuming that the next state is an optimal state[10].

$$U(s) = R(s) + \gamma \max_{a \in A(s)} \sum_{s'} \rho(s'|s,a)\, U(s') \tag{1}$$

In Eq. (1), R gives the reward the agent derives in the current state s and $\gamma \in (0,1]$ is the discount factor, which determines the value that an action taken in the future would yield if taken in the present moment. The higher the value of γ, the higher the importance placed on future expected rewards. $\rho(s'|s,a(s))$ is the probability that the learning agent will transition from the current state s of the environment to the optimal state s' by executing the action a.

The value-based RL algorithms estimate how good or bad it is for an agent to be in a particular state s or to execute a particular action a from the state s by evaluating the expected returns E_π under policy π. Most value functions are expressed according to the format provided by Eqs. (2) and (3).

$$V_\pi(s) = E_\pi\left[\sum_{j=0}^{T} \sum \gamma_j R_{t+j+1}(s,a)\right] \tag{2}$$

$$Q_\pi(s,a) = E_\pi\left[\sum_{j=0}^{T}\sum \gamma_j R_{t+j+1}(s,a)\right] \quad (3)$$

Here, Eq. (2) expresses the value function $V_\pi(s)$ associated with the state s, while Eq. (3) expresses the value function $Q_\pi(s,a)$ associated with taking the action a under policy π from the state s in sub-timeslot t. All the sub-timeslots collectively make-up the timeslot T. Equation (2) is known as the value function, and Eq. (3) is referred to as the Q-function.

Most MARL approaches are model-free approaches. Most of the model-free value based approaches stem from the well-known Q-learning algorithm[9, 11]. The Q-learning algorithm iteratively follows a value-based method to approximate the Q-values of each action in its current state using Eq. (3). In each step of the algorithm, the learning agent selects the action that promises the highest Q-value as the action to take next and as its optimal action policy Q^*, and updates the derived value using temporal difference learning, as shown in Formula (4) below.

$$\begin{aligned} Q_{t+1}^i\left(s_t^i,a_t^i\right) \leftarrow Q_t^i\left(s_t^i,a_t^i\right) + \alpha[r_{t+1}^i\left(s_{t+1}^i\right) + \\ \gamma \max Q_t^i\left(s_{t+1}^i,a\right) - Q_t^i\left(s_t^i,a_t^i\right)] \end{aligned} \quad (4)$$

In Formula (4), $Q_t^i\left(s_t^i,a_t^i\right)$ is the Q-value of the current state s and action a for the i-th learning agent. α is the learning rate, $r_{t+1}^i\left(s_{t+1}^i\right)$ is the reward for the action chosen, and $\max Q_t^i\left(s_{t+1}^i,a\right)$ is the maximum expected future rate. When $\gamma = 0$, the Q-learning algorithm is deemed to be myopic as it is focused only on maximising the immediate rewards and not long-term rewards. However, when $\gamma = 1$, the Q-learning algorithm does not discount the future rewards projected, but instead determines the Q-value using the current reward and all future rewards projected.

Research into RL was initially focused on Single Agent Reinforcement Learning (SARL) approaches where the core focus was to enable agents to learn from their observations of the environment. However, as AI matured, it became clear that independent learning agents operating in the same environment impact each other's learning ability, and limit AI's potential power.

This lack of regard for changes in the operating environment due to competing CRs is evident in the SARL algorithms. Moreover, it has been observed that the SARL approaches cannot be extended to multi-agent scenarios because they are not designed to consider the actions taken by other agents or the effects of these actions on the operating environment before taking further action. MARL approaches must, therefore, enable each CR to independently develop a strategy to optimise its long-term return through rewards or punishments acknowledged from interactions with the environment and the other CRs in the environment[7, 9, 12, 13]. Through this process, the CRs must learn independently and as a collective through trial and error.

MARL approaches are distinctly different from the SARL approaches. Unlike SARL approaches, MARL approaches are required to capture and analyse the agents' interactions with each other and the environment. As such, SARL approaches cannot be directly applied in a multi-agent learning environment. Similarly, the MARL approaches are not suitable to optimally address single agent learning problems[9, 14]. The most apparent differences between the SARL and the MARL approaches are outlined in Table 1[8, 14–16]. Table 1 does not provide an exhaustive review of the differences between SARL and MARL approaches as this is already covered in great detail in Refs. [9, 17, 18].

Despite the vast differences between SARL and MARL approaches, breakthroughs achieved in recent years in SARL algorithms for online video games, robotics, and safe autonomous driving have triggered a burst of approaches to improve MARL algorithms[10, 12–15] and address these challenges.

Similarly, developments in optimisation approaches, game theory, and heuristic approaches are also used to enhance MARL approaches even though these approaches do not facilitate learning[19–22]. Interestingly, most MARL approaches are founded on game theoretic principles.

3 Reinforcement learning from game-theoretic, neural networks, and organic computing viewpoints

MARL approaches are founded on game-theoretic applications[8, 22–25]. Although conventionally studied and applied in economics, political sciences, and biology,

Table 1 Differences between SARL and MARL approaches.

No.	SARL characteristic	MARL characteristic
1	Represent the environment as a Markov Decision Process (MDP) with a specified space and reward function.	Represent the environment as a stochastic game or an extensive form game. The states represent the joint states of all the CRs, and rewards are associated with each joint state.
2	State-action pairs selected by individual CR always yield the same reward whenever selected.	The outcome of the state-action pair selected by a single CR is dependent on the state-action pairs of other CRs operating in the same environment as they collectively make up the joint state.
3	Computational complexity increases as the number of state-action pairs available to the CR increases.	Computational complexity increases as the number of CRs, state-action pairs, and computing episodes/iterations increases.
4	Learning approaches: Direct and indirect.	Learning approaches: Centralised or distributed (co-operative, competitive, and mixed/hybrid)
5	Methods: Model-free, i.e., value-based [e.g., Monte-Carlo, temporal difference (e.g., State-Action-Reward-State-Action (SARSA) and Q-learning)] or policy-based (e.g., gradient/gradient free). Model-based (e.g., dynamic programming and certainty equivalence).	Methods: Game theory (stateless/static games, Markov games, and mean field games), couples with model-free and model-based methods for RL. Heuristics and direct policy search techniques, in conjunction with deep neural networks, have also been applied.

game theory has become a popular framework to model wireless networks (more specifically, resource allocation and management in CR networks[8, 23, 24, 26–28]). Non-cooperative distributed game-theoretic approaches provide a logical approach to model, analyse, and predict the strategic interactions of multiple decision-makers (CRs) without enforcing cooperation. In these approaches, each CR is expected to make choices that maximise its returns (utility) while indirectly achieving collaboration amongst the competing CRs. The players of the game must demonstrate rational behaviour and behave intelligently. These qualities will be demonstrated through the decisions that the CR makes. If the CR chooses strategies that do not improve its utility and instead chooses strategies that are to its detriment, then the game will not converge[11]. The general representation of a game is given by the tuple in Eq. (5) below.

$$\Gamma = \langle \mathcal{N}, \{M_i\}_{i\in\mathcal{N}}, \{u_i\}_{i\in\mathcal{N}} \rangle \tag{5}$$

where $\mathcal{N}$ is a finite set of players and also referred to as CRs. This set of players is assumed to stay in the game for the duration of the game and no new joiners are permitted to participate in the game while it is in progress. M_i is the set of strategies of player i. u_i is the utility (i.e., returns) function of player i. If $\mathcal{N} = \{1, 2, \ldots, N\}$, for brevity, we denote $M = \Pi_{i\in\mathcal{N}} M_i$ as the space of possible pure strategy combinations in the game Γ and $M_{-i} = \Pi_{l\in\mathcal{N}|\{i\}} M_l$ represents the strategies adopted by the opponents of player i. If $M_i \in M$ is the strategy chosen by player i, then each element $\sigma = [M_1, M_2, \ldots, M_N] \in M$ is said to be a strategy profile or set of strategies of the N CRs acting in the shared environment. So, we write $\sigma = [M_i, M_{-i}]$. We also depict the benefit (utility) function for each user $i = 1, 2, \ldots, N$ as a function that maps each possible strategy executed with a real value. That is, each player will derive a utility given by $U_i(\sigma) : M_i \to \mathbb{R}$, and the system will derive a system utility $U_i(\sigma) : M_i \to \mathbb{R}$, where $\mathbb{R}$ is a set of real numbers. Specifically, the utility function is the game's optimisation objective. The utility may be specified based on the parameters the players aim to maximise, e.g., spectrum efficiency. When the CRs reach a point where they continuously play the same strategy, then the system is said to have reached a stable operating point from which none of the CR wants to deviate unilaterally. This stable point is commonly referred to as a Nash Equilibrium (NE).

The NE reached might not be the best achievable NE as some games converge to the first equilibrium point identified which may be a local optimum. In other games, the initial channel/state the CRs start the game from and the order in which the CRs make their decisions determine the nature of the optimum reached. For this reason, many researchers have proposed various approaches to identify the best NE in a game. They also defined numerous other types of equilibria (e.g., the correlated equilibrium, Stackelberg equilibrium, Bayesian equilibrium, and the evolutionary stable strategy) that a game can aim to converge to. Nevertheless, it remains unclear how, in practice, CRs

in a distributed non-cooperative game will converge towards the same equilibrium[17]. For this reason, the problem of ensuring the game to converge to the best equilibrium state remains an open research problem except in cases where the game is formulated as an exact potential game. If the players in a game can observe the strategies executed by all other players, then the players are said to have symmetrical information. Having symmetrical information can aid the game to not get stuck in a local equilibrium, however, information symmetry is not a condition for an NE to be reached[11]. One way to ensure that the game converges to a global optimum and NE, is to define the game as a potential game and the objective function of the game as a potential function.

Suppose the objectives (i.e., the utility functions) of the CRs align to a single objective, the CRs have complete information about the operating environment, and the game always converges to an NE. In that case, the game is labelled a potential game and the objective function is a potential function. Moreover, the NE reached is unique and a global optimum associated with the initial starting point of the CRs in the game. When the strategies selected by the CR in each iteration result in an improvement in the CR's utility and converge to a stable point, the set of strategies is called a Finite Improvement Path (FIP). A potential game wherein each CR has a Finite Improvement Path (FIP) is called a Best Response Potential Game (BRPG)[29]. BRPGs are popular in solving resource allocation and management problems in distributed CR networks because each CR maximises the overall network objective whenever it considers the actions of other CRs and chooses to play a strategy that betters its utility. BRPGs can also be formulated such that a player (CR) does not need to know complete information about the actions selected by all other players (CRs) in the same game. To express the BRPG mathematically, we allow the notation of $\sigma = (m_i, m_{-i})$ to be used in the place of the notation $\sigma = [M_i, M_{-i}]$. Then a BRPG is expressed by the game Γ, if and only if a potential function $U(\sigma): M \to \mathbb{R}$ exists such that $\forall i \in \mathcal{N}$, $\forall m \in M$, and $\forall m_{-i} \in M_{-i}$:

$$\mathcal{B}_i(\sigma_{-i}) = \arg\max U_i(\sigma_i, \sigma_{-i}) = \arg\max U(\sigma_i, \sigma_{-i}),$$

where $\mathcal{B}_i(\sigma_{-i})$ is the best response function of player i given by

$$\mathcal{B}_i(\sigma_{-i}) = \{m_i \in M_i : U_i(m_{-i}, m_i) \geqslant U_i(m_{-i}, m_i') : M\}, \quad \forall m_i' \in M_i, \forall m_{-i} \in M_{-i} \tag{6}$$

In this way, the utility of the action selected by the i-th CR will always return higher returns than any other action it could have selected from its strategy profile. Moreover, the utility returned from the selected action results in the improvement of the overall system utility. The best BRPG converges to an NE defined mathematically as follows, and M^* is an NE if and only if

$$U_i\left(M_i^*, M_{-i}^*\right) \geqslant U_i\left(M_i, M_{-i}^*\right), \forall i, \forall M_i \in M \tag{7}$$

Although the BRPG produces an NE that depicts the strategies that each CR should play to reach a global optimum, it does not develop a reusable policy that each CR can apply to reach the equilibrium state[30]. Moreover, it does not enable the CRs in the game to learn from their experiences of interacting with the operating environment to improve their knowledge of their operating environment and thus improve the decisions they make. Game theory, therefore, is best used as a tool to enable each CR to make decisions depending on the choices of others. However, a potential game, coupled with learning can be a novel way to enable efficient and effective sensing and sharing in multi-CR environments. This is discussed in more detail in Section 4.

To create MARL approaches, RL has adopted game theory's approach to modelling the interactions between multiple agents in a distributed network to reach a set of strategies that will bring the system to a stable point. This infusion enables the CRs to network to reach a set of strategies that will bring the system to a stable point.

(1) Extend the Markov Decision Process (MDP) to a multi-player scenario, sometimes without relaxing the rationality conditions[11, 19, 22].

(2) Assign values to the strategies executed, create a recording (called history) of the experiences experienced, and query the history of experiences to make a rational and informed decision to optimise its utility.

(3) Learn about the operating environment even if the learning agent does not have prior knowledge about

the environment or other agents that are active in it.

At the core of MARL approaches are either the Markov, extensive form, or the mean field games[8, 14, 16, 28]. The Markov and the extensive form games are discussed next.

3.1 Markov games

A Markov game (also called a stochastic game) extends the game represented by Eq. (5). Stochastic games are viewed as matrix games with rewards associated with each joint action[30]. The Markov game is defined by the tuple G given by Eq. (8) below[23].

$$G = \left\langle S, \{A_i\}_{i\in\mathcal{N}}, \{R_i\}_{i\in\mathcal{N}}, T^P, \gamma \right\rangle \tag{8}$$

here, as the case in Eq. (1): the Bellman equation, S is the set of states in the environment and R_i is the utility (i.e., reward/payoff) function of player i because of choosing an action $A_i \in A$ from the current state. Unlike the earlier game definitions expressed in earlier section, the Markov game introduces a new function known as the transition probability function T^P. T^P, as in the case of the Bellman equation, uses information about the actions, $a \in A$, taken by all other players in the environment to control the state transition decision of each player. It is defined as $T^P : S \times A_i \to \rho(s'|s, a(s))$ where $\rho(s'|s, a(s)) \to [0, 1]$ is the probability of the game shifting to a state from its current state by taking the action A_i. γ is a discount factor used to discount future rewards and encourage optimal decision making in the immediate term, as is the case in the Bellman equation. The discount factor in essence models the importance of future rewards in the current state of the game. An illustration of the Markov game action-reward process flow applied in the CR joint sensing and sharing process is provided in Fig. 4.

All stochastic games have at least one solution equilibrium[30]. In Fig. 4, R^{t+1} and S^{t+1} represent the system reward and the joint system state at timeslot $t+1$, respectively. r_i and a_i are the individual reward and action of the i-th player, respectively. Each player's action a_i forms part of the system strategy profile A^t executed at timeslot t. Just as each player's reward, r_i contributes to the overall system reward R^t achieved at timeslot t.

In Markov games, the players can sequentially (i.e., take turns to) decide on actions to take or take

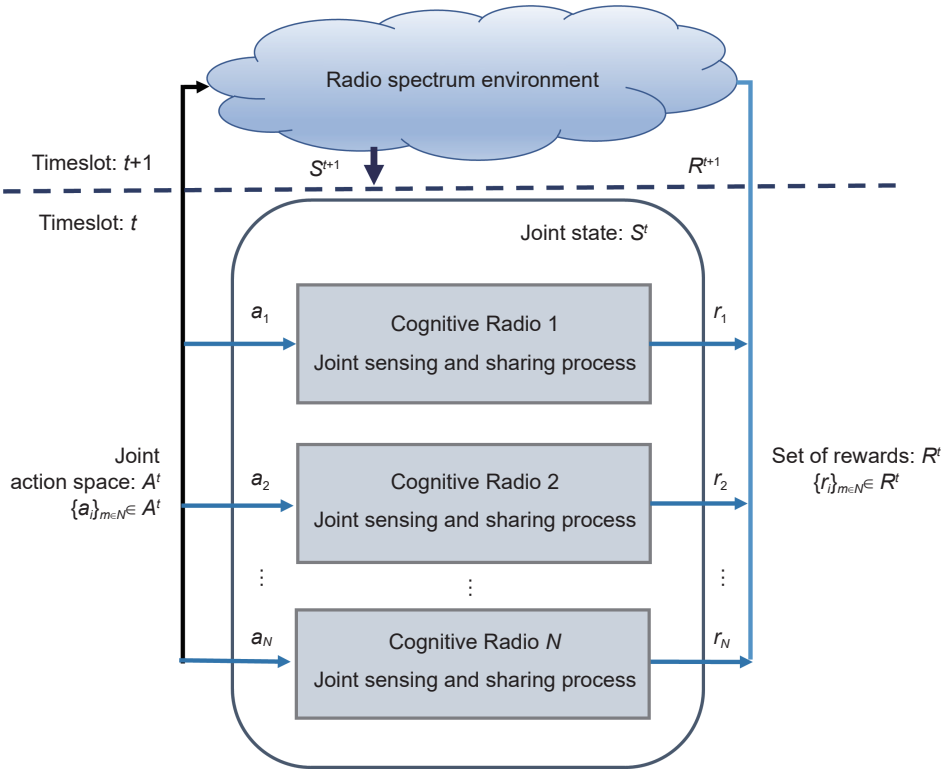

Fig. 4 Action-reward flow applied in Markov games.

decisions/actions simultaneously, considering the actions previously taken[31]. However, each player in the game must be equipped with perfect information (e.g., error-free sensing results)[16]. The requirement for perfect information causes the Markov game to be challenging to compute because it is computationally hard to keep track of the actions of all players and their associated reward functions, especially in an extensive network. This challenge is common amongst all game theory models. This drawback, commonly referred to as combinatorial complexity, is the most evident in large scale networks (network with more than 1000 nodes), wherein the game becomes computationally intractable[32].

The Nash Q-learning algorithm is an early example of an algorithm that aimed to resolve these challenges in a non-cooperative environment. The Nash Q-learning algorithm extends the work done by Littman et al.[23, 31, 33, 34] on combining a Markov game with the single-agent Q-learning method and the minimax method to a multi-agent general-sum Markov game with mixed strategies. The Nash Q-learning approach assumes that an equilibrium exists for every stage game and then repeatedly finds Nash Q-values for each stage game[35]. Other examples of Markovian Game based MARL (MGM) algorithms are outlined in Refs. [35–37] and include variations of Q-learning and Nash Q-learning approaches. The extensive form games are described next.

3.2 Extensive form games

When the Markov game is required to allow players to take actions/decisions sequentially instead of simultaneously, the Markov game is extended to form the Extensive Form Game (EFG)[8]. Unlike the simultaneous moves Markov game, which is described using a tuple, an extensive-form game is best described using a tree representation called a directed graph in mathematics, as depicted in Fig. 5 adapted from Ref. [16]. Mathematically, an extensive form game is defined as $\left(\varnothing \cup \{c\}, \mathcal{H}, A, Z, \{r_i\}_{i \in \mathcal{N}, \tau, \pi^c, S}\right)$, where S, $\mathcal{N}$, and A are defined in the same way as in case of the Markov game. r_i is the reward function for player i and c represents the randomness of chance or nature. c is assumed to have a fixed stochastic policy π^c to specify the randomness of the environment.

The game maintains in $\mathcal{H}$, the history of actions taken to get the CR to its current state. $\mathcal{H}$ is then used to determine the set of actions that the CR can take next, provided the next stage of the game is not the terminal stage. This process is defined as $A(h) = \{a | h_a \in \mathcal{H}\}$, where h_a is the new history formed. $Z \subseteq \mathcal{H}$ represents the set of all paths that lead to the terminal

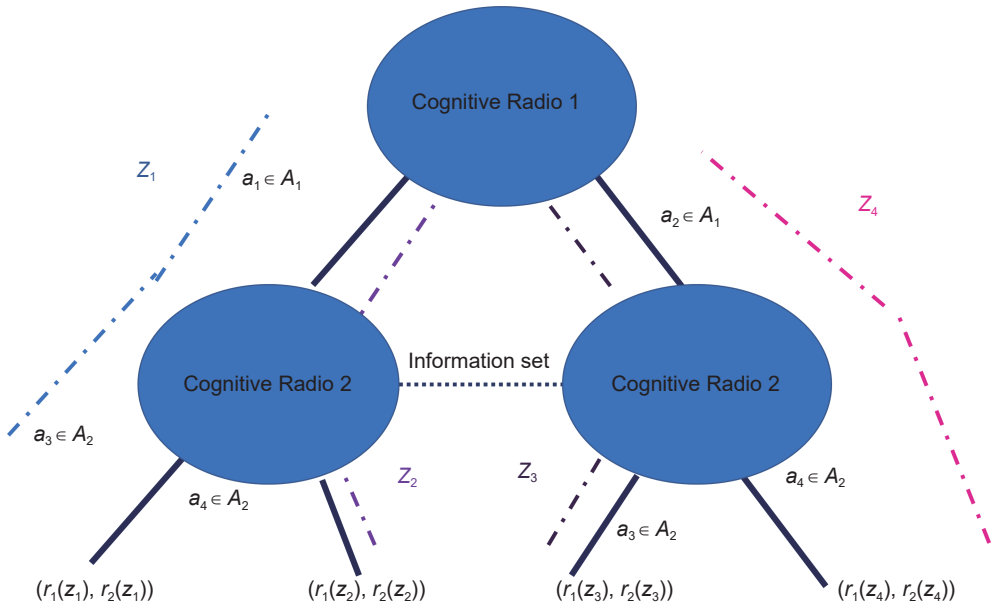

Fig. 5 Decision tree applied in extensive form games, adapted from Ref. [16].

state of the game, and therefore represents the completion of a game. When the game reaches the terminal stage, the reward function is applied to determine the utility derived from the associated path. The reward function is therefore a real value mapped to the terminal histories of the game, that is, $R: Z \to \mathbb{R}$. τ is used in expressing in the game path, and the CR to whom each action stored in the history is associated. It is defined as $\tau: \mathcal{H} \to \mathcal{N} \cup \{c\}$.

The tree representation is preferred in cases where the players must act sequentially, as it depicts the predecessor and successor relationships of the decision nodes (CRs) in the game. Each decision node always has the same information set, which comprises information about the game, player sequence and player sequence status, as well as the actions available and where they lead to. In the game depicted in Fig. 5, Cognitive Radio 1 is the initial node and Cognitive Radio 2 is the terminal node.

There are four paths in this game that lead to the terminal state, namely paths Z_1, Z_2, Z_3, and Z_4. In following the paths, each CR is able to select an action from its strategy profile and recall its previous actions perfectly. An EFG can be defined as a game of imperfect information or perfect information. In an imperfect information game, the CRs are not able to observe each other's previous actions.

The difference in the tree representations of an EFG game played under perfect information conditions and one played under imperfect information conditions lies in how the action-outcome pairs are represented. When the game is played under imperfect information conditions, the probability of each possible action-outcome pair is included in the tree representation of the game. In cases where there are many learning agents and possible states, the combinatorial complexity issue is imminent. Examples of EFG-based MARL (EFGM) algorithms are provided in Refs. [38–40] and include variations of Q-learning algorithms for extensive multi-agent games.

The larger the size of the CR network, the more time prohibitive the EFG becomes because the size of the histories the CRs must traverse becomes large and may cause the CRs to be slow to act[8]. This situation is considered a form of combinatorial complexity. Because EFGs suffer from combinatorial complexity, MARL approaches based on EFGs also suffer from combinatorial complexity. Some algorithms manage combinatorial complexity by enabling the learning agent to traverse a percentage of the history and use this information to decide the next action. An example of this approach is provided in Ref. [39]. Other algorithms convert the game to a Markov game and then apply the fictitious play, actor-critic fictitious play, or fictitious self-play approaches to determine the CR's best response to the actions that it believes the other CRs are likely to take. It formulates its belief of the actions likely to be taken by the other CRs by analysing their historical behaviours. In this way, the CR accelerates the game and reduces the computational time and complexity. Various variations of fictious play algorithms are often applied to achieve this outcome more efficiently and accurately. Although fictious play based approaches date as far back as 1951, their ability to converge is not always guaranteed—especially in games with many nodes.

A popular alternative approach to resolving the combinatorial complexity challenge in EFGM is constructing a multi-stage Markov evolutionary game and applying replicator dynamics[16, 25, 41–43]. As with fictitious play, the replicator dynamics incorporated into the MARL approach aim to pre-empt the actions that are likely to be reselected by a CR's counterparts to accelerate the game and reduce the computational complexity of the game.

Fictitious play based learning algorithms are effective in enabling learning. Replicator dynamics are effective in preempting the actions that are likely to be reselected. However, converting a large EFG to a Markov game or a multi-stage Markov evolutionary game in order to apply fictitious play or replicator dynamics increases the complexity of the game because the newly formed game will have an exponential number of possible actions compared to the expected number of states. An increase in complexity causes an increase in the computational power requirements. These increases are detrimental to the battery life of a CR[42]. Therefore, the

computational and time efficiencies gained through fictious play and replicator dynamics are lost through the restructuring of the game. Secondly, these algorithms require a lot of computational memory to store each player's increasing history and a lot of processing power to traverse each of these histories. High computational memory and processing power requirements are challenging to satisfy in small and battery powered CRs.

Other challenges experienced by Markov game based MARL and EFGM algorithms relate to complex rationalisation, non-optimal learning outcomes, algorithm intractability, privacy, and security[16, 17]. The complex rationalisation challenge is caused by rationalising multiple learning objectives. The non-optimal learning outcomes challenge occurs due to uncertainty regarding the optimality of the learning outcome achieved. It also relates to the game converging to a non-optimal NE, as is the case in games that are not formulated as potential games. Finally, algorithm intractability, privacy, and security challenges are caused by an increase in the number of players in the network[14, 16, 24].

The combinatorial complexity, non-optimal rationalisation of the game objectives, and the non-convexity of the objective functions of the games make the process of finding a pure strategy NE in a multi-agent learning environment a PPAD-complete, PSPACE-hard, or an NP-hard problem depending on the time-horizon set, even in two-player general-sum games[17, 18]. Moreover, this process is time-consuming and computationally demanding. Therefore, finding the most optimal NE by following a brute force or exhaustive search approach would exacerbate the time consumption and computational resource requirements.

Many researchers have made contributions to this challenge using Multi-Agent Trust-Region Learning (MATRL) algorithms[44], convex games, and quasi-convex games, however, most of the results are either theoretical (untested or unsimulated) or still suffer from the challenges expressed above. As a result, researchers are turning to mean field game based MARL and data classification learning methods, more specifically, the supervised learning methods such as the Artificial Neural Networks (ANNs) and deep neural networks to resolve these challenges.

MARL approaches that leverage deep learning techniques are commonly referred to as Deep Reinforcement Learning (DRL) approaches. These are presented in the next section.

3.3 Deep reinforcement learning

Early attempts to address long convergence time, high resource requirements, and resource requirements related challenges in MARL approaches did so by combining Q-learning (and various other model-free algorithms) with non-linear approximations of the value function or non-policy learning approaches, however, these combinatorial approaches did not converge[45]. The next iteration of approaches replaced the non-linear value-function approximations with linear value or policy function approximations. Although these approaches would converge, their convergence time was still high, rendering them non-optimal[46]. Finally, researchers directed their attention to applying Artificial Neural Networks (ANNs) to approximate value and policy functions. An example of such an approach was proposed in Ref. [44], where the learning agents find the region within which the most optimal NE exists, viz., the trust region, by coupling their trust-region learning method with a meta-game analysis that improves learning stability and efficiency using an ANN.

Unlike previous function approximators, ANNs provide a structure to store and update the various parameters associated with each action that an agent can select. The benefit of ANNs lies in that the agents do not require complete knowledge of the action and state space for the algorithm to work. This benefit is particularly useful when the agent is confronted with a large state and action space. An ANN with multiple layers between the input and output layers is called a Deep Neural Network (DNN). DNNs, as with ANNs, serve as universal approximators[17]. However, DNNs can model complex non-linear relationships even with incomplete information about the environment. A

remarkable breakthrough in applying DNNs in RL was outlined in Ref. [47], where the novel deep Q-network method for SARL algorithms is introduced. The contribution made in Ref. [47] is marked as the birth of DRL.

When applying the multi-agent DRL approach to a distributed CR network, each CR uses its DNN to approximate its optimal utility functions. To do this, they use the system's current state to learn to predict the action that will yield the highest reward for the CR in the next step or to identify the action that has the highest probability of yielding the desired outcome. The CR then observes the reward received from the environment and updates its DNN accordingly[17].

In Ref. [48], it is shown that DNN overlayed over traditional RL approaches results in that algorithms fall into the same model-free or model-based category that the initial RL algorithm, before it was coupled with the DNN, fell into. This is because the DNN is used to train the RL approaches and therefore does not alter the categorisation of RL categories. However, we have observed the development of a new category of RL algorithms which are characterised as the fusion of model-free and model-based algorithms. These infused algorithms form a new category of RL algorithms called the quasi model based RL category. This new category of RL algorithms is depicted on Fig. 6, which was adapted from Ref. [21] and expanded to depict the shift towards model-based and model-free infused approaches.

DNNs are applied mostly in the model-free RL algorithms as adaptive function approximators because they allow the learning agent to process high dimensional state and continuous action inputs and to learn relevant feature representations and policies[48]. However, unlike RL, ANNs fall in the machine learning category of supervised learning and require training to classify the inputs received correctly. Training data can be time consuming and may result in the CR losing the identified spectrum opportunity. More specifically, when DNNs are applied in RL, the problem of non-i.i.d. (not independently and identically distributed) and non-stationarity of the training data and function estimates that diverging from the optimal value function is prevented[49]. Non-i.i.d data result from the training data comprising highly associated sequential agent-environment interactions, violating the independence condition. In this way, the DRL algorithm addresses the non-stationarity problem in the action-state space associated with the Markov game MARL

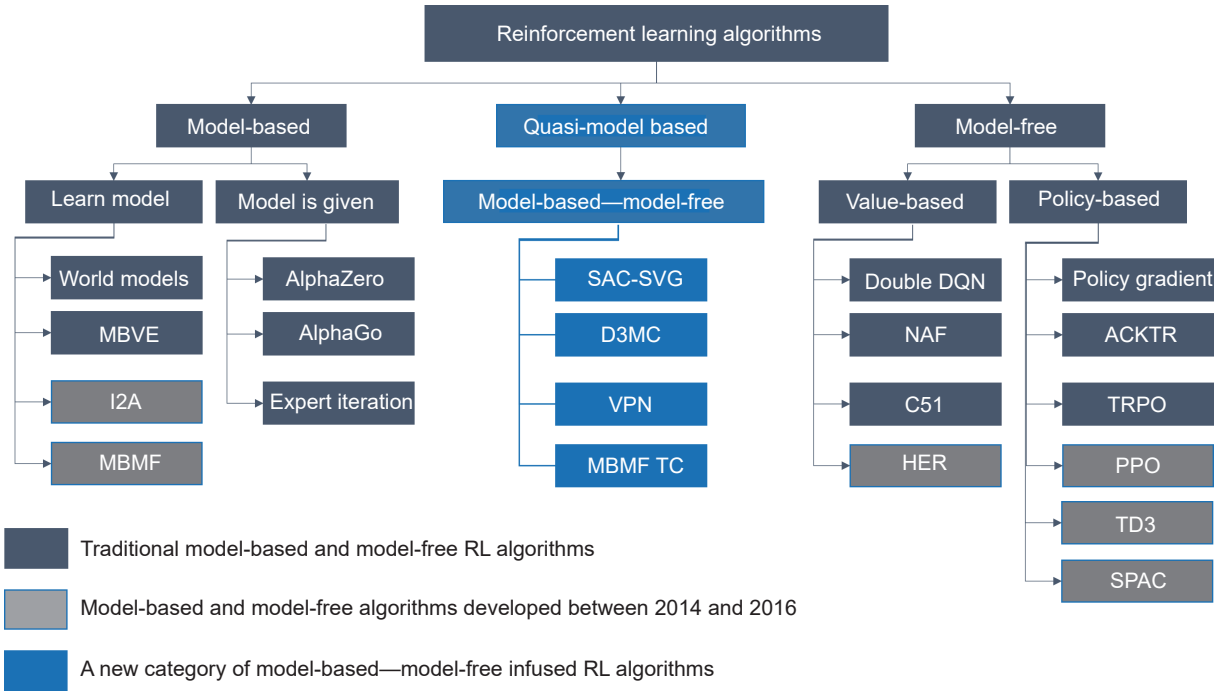

Fig. 6 Classification of newly developed RL algorithms, adapted and expanded from Ref. [21].

and the EFGM. However, it re-introduces non-stationarity in the training phase because the training data distribution takes a non-stationary form as the agent continuously learns its operating environment[49].

The non-stationarity challenge causes poor design choices—better understood as poor function approximator representation decisions—leading to divergent function value approximations. It also causes poor reproducibility of theoretical results in practice. These challenges are discussed in more detail in Ref. [49].

Traditionally, gradient-based algorithms such as the backpropagation algorithm and the stochastic gradient descent algorithm were used to train DNNs. Werbos[50] invented the backpropagation training algorithm in 1975, and the stochastic gradient descent method could be linked to the work done by Robbins and Monro[51] in their paper, A Stochastic Approximation Method. In recent years, non-gradient evolutionary algorithms, specifically gradient-free and population-based Genetic Algorithms (GAs), have been accepted as plausible DNN training methods. These three training methods have been incorporated into many DRL approaches as outlined in Refs. [45, 48, 52].

An analysis of an arbitrary set of recently developed DRL algorithms shows that most algorithms configure the training algorithm to allow the DNN to be trained using a centralised critic. At the same time, the learning process remains decentralised for each actor[53]. This framework is commonly referred to as the Centralised Training Decentralised Execution (CTDE) framework. It comprises centralised value function methods and Value Decomposition (VD) methods[54] to approximate the value function. CRs applying the VD method learn the various action-value functions of the individual CRs in the network to estimate the joint action-value function, whereas CRs applying the centralised value function method use models estimated through their policy learning procedure to learn approximate models of other agents in the network[54].

Most value-based MARL approaches are designed following the CTDE framework. The framework is best suited for cooperative MARL approaches[54], even though it can be applied in decentralised approaches. When applied in decentralised networks, training is often limited to a single CR or a subset of CRs, while learning is always decentralised. In addition, CTDE suffers from the curse of dimensionality, non-stationarity, and depleted space/memory when applied to vast networks[49]. Consequently, CTDE framework based DRL approaches, in the absence of factors to mitigate against the cited pitfalls, are considered impractical for use in large scale networks such as those found in beyond 5G networks[55]. Hence, the attention-based actor-critic algorithms and the Decentralised Training Decentralised Executing (DTDE) actor-critic framework based algorithms are gaining popularity as suitable replacements of the CTDE framework applied in some DRL approaches[53, 55].

There are several recent works that are considered significant contributions in the field of MARL and more specifically in DRL, because they have attracted a lot of research attention and many variations to the first contributions have since been made. These algorithms are depicted in Fig. 6, taken and expanded from Ref. [21]. Examples of these notable contributions include the Deep Deterministic Policy Gradient (DDPG), twin delayed DDPG, Hindsight Experience Replay (HER), Imagination-Augmented Agents (I2A), Model-Based RL with Model-Free Fine-Tuning (MB-MF), Proximal Policy Optimisation (PPO) algorithms, and the Self-Play Actor-Critic (SPAC) algorithm. Further detail about these algorithms is presented in Refs. [17, 18, 45, 56−64]. A high-level comparison of these algorithms is presented in Table 2.

The algorithms described in Table 2 have made a significant contribution in terms of entrenching deep learning in RL algorithms. Through deep learning, these algorithms have broken time and computational complexities to

(1) address the disadvantages of model-based DRL training approaches such as the sample inefficiency, overfitting challenges, and algorithm instability;

(2) reduce the complexity of the algorithm by optimising the inputs taken and improving the algorithm wall-time and the algorithm performance;

(3) present algorithms that provide better performance than possible through traditional algorithms

Table 2 Notable DRL algorithms developed since 2014.

Method summary	Key feature	Computational complexity
Hindsight Experience Replay (HER)[56]: A data training algorithm that enables model-free RL algorithms to mimic the ability of humans to learn almost as much from achieving an undesired outcome as from achieving the desired one, i.e., enables the agent to learn even when the reward signal is sparse or binary. In this way the MARL objective is not singular, and training is decoupled to enable insights from unintended outcomes to be used to direct wanted outcomes.	● Rewards are granted based on the outcome achieved and not the initial goal set. ● Experiences are replayed with different goals to expand the learning gained by the agent and recall rare but probable occurrences. ● Algorithm can be applied over a DQN and in any model-free MARL algorithm.	● Defining goals and setting the optimisation approach are a difficult process. ● Require a lot of memory and computational processing power. ● The run time can be long depending on the variation of the HER algorithm applied.
Imagination-Augmented Agents (I2A)[63]: A data training algorithm that aims to form imagination augmented RL approaches for approximate environment models. It models the environment by initialising an imagined trajectory using the present time real observation and subsequently feeding the simulated observations into the RL model. It uses predictive analysis and imagination to enable the learning agent to create implicit plans and take advantage of model-based and model-free RL jointly, without the pitfalls of regular model-based approaches. The predictive analysis is conducted over predictions obtained from the environment.	● Augment model-free RL agents with imagination to enable them to construct implicit plans or policies. ● Improve the learning algorithm's performance. ● Demonstrate a superior ability to interpret imperfect predictions even in unknown environments. ● Use predictions as an additional context in developing the DQN.	● Need a moderate number of iterations to become efficient, however, too many iterations result in diminishing returns. ● Moderately complex to compute due to its minimal reliance on the model of the environment and increased focus on its predictive ability. ● Perform slower than model-free RL approaches.
Proximal policy optimisation algorithms[62]: A data training algorithm that alternates between sampling data through interaction with the environment and optimising a "proxy" objective function that enables multiple epochs of minibatch updates using stochastic gradient ascent. The method is applied over RL approaches to improve their efficiency.	● Leverage a dynamic learning rate so the algorithm can self-correct through the learning rate when the common pitfalls of policy gradient methods resurface (e.g., inconsistent policy update and high reward variances). ● Reuse samples more than once to mitigate against the sample inefficiencies challenge prevalent in traditional policy gradient methods.	When compared to the traditional policy gradient method, this method has a simplified implementation process, decreased sample complexity, improved learning performance, and improved algorithm convergence-time.
Model-Based RL with Model-Free Fine-Tuning (MB-MF)[61]: Use a moderate number of samples and medium-sized neural networks to leverage the benefits of model-free DRL algorithms and model-based DRL in a joint approach. The two approaches are combined to produce stable and conceivable steps for an agent to conduct complex tasks well.	● The algorithm initialises a learning agent using model-free RL combined with DNN features while also applying a medium-sized neural network model combined with Model Predictive Control (MPC) over a model-based RL algorithm.	● When compared with model-based or model-free fine-tuning, MB-MF has increased performance, reduced complexity, and improved sample efficiency over a wide range of complex tasks.
Twin Delayed Deep Deterministic (TD3) policy gradient algorithm[57]: Designed as a model-free, online, and off-policy reinforcement learning method that extends the DDPG by relating the target network bias to the over-estimation bias, using the minimum value between a pair of actor-critics, and delaying policy updates.	● Improve the learning speed by applying two Q-value functions. ● Minimise the effects of function approximation errors on both the actor and the critic by using the minimum value function estimate during policy updates. ● Prevent overestimated value estimates and sub-optimal policies by adding noise to the target action, which makes the policy less likely to exploit actions that have high Q-value estimates.	● When compared to the DDPG, the approach has improved learning speed and minimal estimation biases and approximation errors. ● Suffer from slow convergence. ● Long training duration. ● Prone to converging to a local optimum. ● PPO often outperforms TD3.
Self-Play Actor-Critic (SPAC)[60]: Combine a wide-ranging critic into the policy gradient method to form a self-play actor-critic with imperfect information.	● Improve stability and sample efficiency of the self-play reinforcement learning training procedure. ● Usable in environments with limited information. ● Speed up the training process.	● Increased algorithm performance (outperform DDPG and PPO). ● Reduced sample complexity. ● Improved sample efficiency over a wide range of complex high-dimensional tasks.

by combining model-free and model-based actor-critic algorithms.

However, DRL algorithms continue to grapple with long convergence time, converging to a local maximum, and high resource requirements that must be satisfied to facilitate learning, specifically in CRs. Some of the quasi model based algorithms outperform the algorithms described in Table 2, however, the field is still being developed and tested.

In recent years, a connection has been drawn between Mean Field Games (MFGs) and MARL to form the field of Mean Field RL (MFRL)[65, 66]. MFGs are discussed next as they are not prone to as many challenges as DRL approaches, even as the number of agents becomes very large.

3.4 Mean field reinforcement learning

MFGs are a class of Markov games played by many homogeneous players, but with the joint action space reduced to what looks like a two-agent joint action space. In this game, each player considers its best response action to the mean effect of the actions taken by neighbouring/other agents in the operating environment. In this way, players do not have to concern themselves with the individual actions taken by other players in the network. Players also do not need to know the transition probability function. MFRL approaches are gaining popularity because of their ability to converge to an approximate NE. MFRL games converge to an approximate NE because the underlying MFG converges to a Mean Field (MF) equilibrium (also known as an ϵ-NE)[57]. Similarly, MFRL approaches do not suffer from the curse of dimensionality, and the complexity decreases as the size of the network increases[48, 67]. Moreover, the impact of a single player's actions on the mean effect of the actions taken by all other players diminishes as the size of the network (number of players) increases, making the impact of non-stationarity negligible in MFGs[68]. Although there is a vast amount of research into MFRL, we limited our review to MFRL approaches for non-cooperative distributed networks. We found that most of these studies were produced between 2016 and 2022 and ranged from classes of approaches to algorithms that aim to solve specific use cases using MFGs combined with model-free RL algorithms and, in a few instances, using MFGs combined with model-based RL techniques.

MFGs have been successfully combined with the Nash-Q-learning and actor-critic algorithms to form the MF-Q algorithm proposed in Ref. [66] and thereby enable MARL in large scale networks. Actor-critic fictitious play, fictitious play, and fictitious self play approaches are also suitable for MFGs, as was shown for Markov games. The interested reader is referred to Refs. [16, 69, 70] for examples of fictitious play applications in MFRL algorithms.

The first instance of the MF-Q algorithm used a mean field approximator to approximate the Q-function and, like the Nash-Q algorithm, converged to an NE. A theoretical proof of this convergence was provided in Ref. [66] for the interested reader. Likewise, subsequent versions of the MF-Q algorithm proved that the mean field approximator could be replaced with a universal function approximator such as the DNN and still converge to an NE[66, 71]. As shown in Ref. [71], replacing the mean field approximator with a neural network approximator converts the MF-Q algorithm to a deep MF-Q algorithm suitable for application in deep MFRL.

More recent studies have started to bring into question the validity of the single mean field, also commonly referred to as the second virtual agent, specifically in cases where the agents are not homogenous. In such cases, the agents could have different objectives or abilities[68, 72]. In Ref. [68], another notable variation of the MF-Q algorithm, referred to as the MFG with best response learning dynamics algorithm, is presented. This algorithm generalises the MFG to enable players to have different states. It further enables the players to learn the transition probabilities of the other players using a posterior sampling approach and allows a player to follow an oblivious strategy that directs the player to select an action considering only the state it is in. A theoretical proof that this algorithm converges to an MF equilibrium despite allowing each player to

disregard the actions and Q-values derived by other players is available in Ref. [68].

Reference [72], in addition to providing additional context into why it is essential to consider the differences in the players, proposed two algorithms to address these differences in MFRL. The first algorithm is the Multi Type Mean Field Q (MTMFQ) learning for known types, while the second algorithm is the multi type mean field Q-learning for unknown types. The multi type mean field Q-learning for unknown types of algorithms is like the first algorithm. However, it begins by applying a K-means clustering approach to cluster the players into known types, as far as is possible. Once the unknown player types have been clustered into the known types, the algorithm reduces to the MTMFQ algorithm. Although these algorithms provide more accurate results than the standard MFRL algorithms, such as the MF-Q algorithm, they are much more computationally expensive than the MF-Q algorithm despite not catering completely for heterogeneous players[70].

Along with these studies, the field of Stationary MFGs (SMFGs) based RL approaches is also being expanded. A sound basis for this field is provided in Ref. [67], where Subramanian and Mahajan presented a case for generalising MFGs to SMFGs for RL as a way to reduce complexity and achieve improved convergence time. These performance improvements are attributed to the replacement of the time-varying policy and the time-varying mean field used in MFGs with a single policy and a single mean field in SMFGs.

Another developing area of interest is in the management or reduction of performance loss in MF-MARL. In Ref. [73], an approach was proposed to minimise and quantify the policy's regret (i.e., loss in performance) due to effects from multiple sub-systems or objectives in an MF-MARL environment. The regret is minimised considering a central and independent agent that knows the system model under which the MF is formulated. Although the simulation results proved the approach's effectiveness, further investigations are required to assess the effectiveness in vast networks with a sizeable action-state space.

Developing alongside MF-MARL is the field of Organic Computing (OC). A high-level overview is presented next as it is foreseen that a link between OC and MF-MARL may be developed to address some of the open questions and challenges presented in Section 3.5.

3.5 Organic computing

Ubiquitous Computing (UC) is more prominent in the 21st century as more and more computing devices, systems, platforms, and computing approaches (each with extraordinary computing power, exceedingly complex designs, and a multitude of interacting components) interact with each other[74]. However, the advent of UC has exposed the need for control over computing capabilities as system designs increasingly fail to foresee and articulate all possible and permissible operations that a computing capability can execute, thereby giving rise to the field of Organic Computing (OC)[74], which was first introduced in Ref. [75]. Therefore, OC aims to enable system designers to apply new design principles to set goals for the computing capability to achieve and enforce control over these complex computing capabilities. In OC, control is enabled by introducing life-like or more specifically, human-like operations, referred to as self-x properties that the computing capability can utilise to realise the set goal. A computing capability is said to possess self-x properties if it can self-organise, self-configure, self-repair, self-protect, or adapt to the environment as required. For system designers, this means that the designs need to be altered from specifying low-level parameters and controls to specifying goals to be met[76]. A further parallel can be drawn between OC, automatic computing, and the capabilities envisioned for the CR. In this way, learning algorithms are required for self-x properties to be enabled in the field of OC.

Unlike game-theoretic MARL, which learns the strategic interactions of players with each other and with their environment, OC aims to learn the interactions and perceptions developed (through sensors and actuators) between an autonomous sub-system and its operating environment[13]. As such, the current success of OC is restricted by the principle of bounded rationality. Bounded rationality specifies that

the computing capability's ability to learn and improve its experiences is informed by the capability's own sensed view or observations of the environment[13, 77]. Nevertheless, as discussed in Ref. [78], this presented an opportunity to integrate MARL with OC to improve the learning potential of automated and connected systems. Evidence of this opportunity was presented in Ref. [79], wherein a form of online RL is incorporated into an OC-based algorithm designed to enable self-organisation and self-adaptation properties. However, there is still limited integration of RL in OC because OC algorithms cannot enable self-x properties if they are dependent on given sources of knowledge (such as a utility function) and sample behaviour data (such as DNN training data)[80]. Moreover, the DRL compute-time must be optimised further if they are to be incorporated in OC algorithms as the existing DRL algorithms are slow to converge and would therefore result in a slow reaction time for the OC algorithm.

MARL algorithms are essential in CR networks as they provide a concrete step toward enabling autonomous learning in unfamiliar environments. Autonomous learning in CR networks is vital to enabling opportunistic access to the spectrum. Allowing opportunistic access to spectrum enables secondary users, like CRs, to gain access to licensed spectrum when the licensed user (PU) is not using the spectrum. This process not only aids in bridging the digital divide but also helps to make the vision of having multiple networks sharing the same spectrum, as is envisioned in ubiquitous networking, become a reality[3].

The CR utilises its cognitive engine to conduct spectrum sensing and sharing processes. These processes produce insights that inform the CR of vacant spectrum it can access and how it should reconfigure its parameters to use the spectrum on its own or with other CRs that may already be active on the spectrum. However, the CR is battery-powered and small and thus has limited computational power and memory. Yet, the CR spectrum sensing and sharing processes are interdependent and executed iteratively. Each timeslot must begin with spectrum sensing (to ensure the licensed user has not resumed activities on the spectrum), followed by the sharing process, and after that, conclude with data transmission, if it is feasible to do so from a time and resource perspective. Therefore, these two processes are time sensitive—they must be concluded very quickly so there is sufficient time remaining for data transmission. In this way, the CR's resources will be used optimally and will remain available for longer. This implies that the learning process must be time and resource efficient. That is, the CR must learn to avoid selecting highly contested spectrum when there are other spectrum opportunities, as this will result in the CR missing transmission opportunities or transmitting at a non-optimal rate. Similarly, the time it takes to conduct the sharing process can be significantly reduced by allowing the CR to use its past sensing and sharing experiences to determine how to reconfigure itself. Therefore, if learning can be used to improve each process, the joint process can become more efficient. However, if learning is applied only to one process and not to the other, then efficiencies gained in one process will be lost in the following process.

Although suitable for use in unfamiliar environments, with or without a model of the environment, MARL still has long run-time and, thus, high resource requirements. For this reason, we have developed a novel MARL approach for joint sensing and sharing in distributed CR networks. This approach is presented in the next section.

4 Multi-agent reinforcement learning approach for joint sensing and sharing in cognitive radios

In distributed non-cooperative CR networks, each CR relies on the accuracy of its spectrum sensing results to decide on the occupancy status of channels sensed. The sensing decisions of the CR must enable the CR to avoid missing spectrum opportunities that exist and avoid using spectrum that is being used by the Primary User (PU). In this way, the CR avoids causing interference for the PU. The sharing process includes the power control task and it is executed following the sensing decision. The transmission power the CR decides to use must enable the CR to maximise its

throughput over the identified spectrum or preserve its resources and forgo the opportunity, if it is best to do so.

Early applications of MARL in the context of CR networks were based on the model-free game-theoretic MARL approaches described in previous sections[81]. Model-free applications were adopted because of the non-stationary nature of the CRs' operating environment. These applications would generally specify the state of a CR as a combination of the channel the CR was transmitting from and the transmission power used for the transmission. The action would be specified as the channel the CR is switching to and the associated transmission power setting the CR will use to transmit over the new channel. The reward for the CR would be informed by the following:

(1) presence or absence of the PU or other CRs on the channel the CR has shifted to;

(2) transmission power level (i.e., was it below or above the pre-set threshold);

(3) noise-levels of the channel;

(4) the outcome of the transmission (i.e., was it successful or unsuccessful due to a collision or environmental factors).

However, these MARL applications suffered from the curse of dimensionality, and inadequate space to store the state-action values as the size of the network became large. This approach was customised in later iterations by replacing the value approximation function. As an example, the value approximation function was replaced in Ref. [82] with the Kanerva-based function approximation. Although performance improvement was evidenced, the algorithms were still not suitable for large scale networks. In other algorithms, the value approximation function was replaced with DNNs. A notable contribution was presented in Ref. [4], wherein a DNN is used as a function approximator to enable learning in a partially observable, multi-CR, and multi-channel environment using the Double Deep Q-Network (DQN) method. The experimental results show that the algorithm converges to a stable point even though information sharing and transfer learning were not enabled. However, the experimental results are limited to small scale networks and the sensing process is not optimised. Other applications of DRL in CR networks are discussed in Refs. [20, 83].

Some of the DRL approaches presented in earlier sections have been applied in distributed CR networks, such as the PPO, TRO, DDPG, and various actor-critic approaches (namely the A2C and the A3C algorithms). Details about these implementations are discussed in Ref. [20]. However, the focus of these algorithms was not on joint sensing and sharing. Instead, these existing algorithms would need to be modified in the areas specified below to be successfully used to address the problem of joint sensing and sharing in large scale CR networks[20, 83].

(1) Introduce complete or quasi-complete knowledge of the operating environment, including accurate channel models and real-time Channel State Information (CSI).

(2) Ensure computational tractability by improving the computational efficiency and performance practicality of the DRL algorithms. This includes replacing the CTDE applications with the DTDE framework in the DRL approaches and balancing the data quality with the required learning objective.

(3) Mathematically formulate the joint sensing and sharing optimisation problems to prevent intractable optimisation or convergence to a local optimum.

(4) Enable the algorithms to cater for non-stationarity or fast-moving CRs and PUs, and thus non-stationary operating environments.

(5) Modify the CRs to form super-computers suited to handle the computational complexity associated with large scale/dense networks. This modification would need to be innovative so as to alter the CR from being a small battery-powered device.

In addition to the modifications listed above, we have observed that other reasons why multi-CR RL approaches are developing at a slow rate are the varying levels of complexity introduced by the various CR network types, learning objectives, and network architectures applied. Although training algorithms (such as the algorithms described in Table 2 and quasi model based approaches depicted in Fig. 6) and MFRL

approaches (such as the MTMFQ algorithms) have not, as far as we have read, been applied in CR sensing and sharing, we believe such applications will reduce the time duration and computational complexity associated with the data training aspects of the problem. However, there is still a need for a multi-CR RL approach to address the need for efficient inference after training has occurred. We therefore propose a novel multi-CR RL approach that can be coupled with the above DRL training algorithms to optimise the training and the learning aspects of the joint sensing and sharing problem.

A system model comprising of a distributed CR network of N CR transceiver-receiver pairs which are randomly distributed in an area with radius r, and a wideband channel with K non-overlapping subchannels is considered. The Physical (PHY) layer specification of the wideband channel is assumed to be based on Orthogonal Frequency Division Multiple Access (OFDMA) for both Upstream (US) and Downstream (DS) access, as is the case in LTE and Wi-Fi standards. Each CR transceiver-receiver pair is focused on identifying a vacant subchannel that best meets its transmission requirements. A subchannel is considered vacant only when the PU is not active on it. The CR transceiver-receiver pairs are assumed to be stationary or slow moving to enable the system to achieve a steady state. Each CR conducts energy detection sensing over its candidate channel to determine the occupancy status of the channel. Each CR senses at most one channel at a time as it is equipped with a single antenna. All the CR transceiver-receiver pairs have equal priority to access the channels. All channels are assumed to have equal bandwidth; however, a transceiver can select only one channel at a time through which to transmit its data. The CR transceiver-receiver pairs cannot transmit and receive data simultaneously.

The primary and secondary (CR) networks have perfectly synchronised timeslots. Each timeslot has a fixed duration T used by the CRs for spectrum sensing and data transmission. The Primary User (PU) is assumed to have time slotted access, that is, the PU occupies channels at the beginning of the sensing time slot or remains idle for the duration of the time slot. The CRs are required to decide on the occupancy status of the channels and the transmission power to utilise over the channel selected for use. The CRs do not share their channel occupancy decisions, but they broadcast their transmission power settings. The time associated with this information sharing process is negligible. A list with the important symbols introduced in subsequent sections can be found in Table 3.

We assume that although each CR is enabled to make simultaneous or sequential decisions as the need arises and to align the system model to the learning process found in nature[10], in this game the CRs make sequential decisions. Moreover, each CR has an objective to optimise its utility. However, the utility function for each CR $i = 1, 2, \ldots, N$ is the same and it is represented by the potential function depicted by Eq. (9).

Table 3 Symbols used in Markov potential game.

Symbol	Meaning
N	Number of CR transceiver-receiver pairs.
K,β	Number of Additive White Gaussian Noise (AWGN) subchannels and subchannel bandwidth.
T,T_s,B	Frame duration, sensing duration, and memory B.
τ,λ	Sensing decision threshold and test statistic.
H_0,H_1	Hypothesis that the subchannel is vacant and hypothesis that the subchannel is occupied.
$\mathcal{P}(H_0)$	The probability that hypothesis H_0 will prevail.
G,G_{ijk}	Channel gain matrix and channel again between the i-th transmitter and the j-th receiver over channel k.
$\varsigma_{ik},\varsigma_{\min}$	Actual SINR of the i-th CR on channel k and minimum SINR required for successful transmission.
n_0,σ_s^2	AWGN with mean zero, variance σ_v^2, noise uncertainty coefficient α, and signal variance σ_s^2.
A,A_{ik}	Channel selection matrix. It depicts if channel k is selected by the i-th CR. $A_{ik} \in [0,1]$.
P,P_{ik}	Transmission (Tx) power matrix takes the value of the i-th CR's Tx power estimation for subchannel k, if the estimation is positive, otherwise it takes value 0.
U,U_{ik}	Utility matrix. Entry $u(i,k)$ takes value of the i-th CR's utility estimation over channel k.
$P_{\min}$	Threshold that the interference generated should not exceed to preserve the CR receiver's decoding capabilities.
r_0,ρ	Scaling factor corresponding to the Fraunhofer distance. Spatial density of the nodes; A positive constant dependent on the number of resolvable paths and their variances.

$$U_i(a(i,k),A_{-i}) = \begin{cases} \left(1-\frac{T_i}{T}\right)\left(1-\rho_k^{\mathrm{FA}}\right)\sum_{k=1}^{K}\mathcal{P}(H_0^i)C_{ik_2}, \\ \text{if } a(i,k)=1 \text{ and } a(i,m)=0, \forall m \neq k; \\ 0, \text{ if } \sum_{m=1}^{K} a(i,k)=0 \end{cases} \tag{9}$$

for $i \in \mathcal{N}, k \in K$ channels. $a(i,k)$ represents the channel selection k of player i. ρ_k^{FA} is the probability of false alarm, $T>0$ represents the duration of a timeslot, and C_{ik} represents the channel capacity.

The capacity C_{ik} is expressed using Shannon's capacity formula, and it is given by Eq. (10). The channel capacity C_{ik} that a CR can achieve is dependent on the Signal to Interference Noise Ratio (SINR) ς_{ik} of the channel selected for transmission.

$$C_{ik} = \beta \log_2[1+\varsigma_{ik}] \tag{10}$$

where

$$\varsigma_{ik} = \frac{P_{ik}|G_{iik}|^2}{\sum_{j\neq i}^{N} A_{ik}P_{jk}|G_{ijk}|^2 + n_0}, A_{ik} \in [0,1] \tag{11}$$

In addition, for each channel, there exists a minimum SINR denoted by $\varsigma_{\min}$ which results in successful transmission. The spectral efficiency projected by the CR must exceed the channel capacity associated with the minimum SINR, failing which, the CR should refrain from transmitting.

Given that multi-agent environment is stochastic, we formulate the game as a Markov potential game Γ_{MP} expressed by Eq. (12) below. Further details regarding the formulation and solution of Markov potential games in a multi-agent environment are provided in Ref. [18].

$$\Gamma_{\mathrm{MP}} = \left|(Q,\{D_i\}_{i\in N},\{U_i\}_{i\in N},T^p,\gamma_d)\right| \tag{12}$$

where Q is the set of states in the game, D_i is the pure strategy channel sensing and access decision made by the i-th CR, and U_i is the utility derived by this CR as a result of D_i. It is also important to define D_{-i} the pure strategy channel sensing and access decisions of all other CRs and $D = \Pi_{i\in N} D_i$ as the space of possible pure strategies combinations in game Γ_{MP}. T^p is the transition probability function and γ_d is the discount factor. The CRs use the approach presented to determine their individual best response strategy and learn from their experiences to improve their future decisions until they reach an NE or terminal state.

Through the approach provided, the CRs are equipped to combine the advantages of game theory using the potential game structure. Convergence to a unique pure strategy stable point is also guaranteed. This is important because many learning algorithms do not converge because all the players in the environment learn through trial and error and thus causing their environment to change, while considering their environment as static. In such algorithms, the learning process does not reach a stable point because the CR continues to update its policy as the learning environment changes. The learning environment is changed by the actions of other CRs operating in the same environment. The outcome of our proposed approach, although not simulated, is expected to extend beyond the stable point reached. It includes a process of learning how the stable point is reached. In this way the CR can improve its channel sensing and sharing decisions in subsequent iterations.

Moreover, the proposed approach can be implemented following a modularised framework with three main modules namely the training, best response process, and the RL process. Following this design will enable the substitution of the training section of the approach with any preferred DRL training algorithm such as the algorithms presented and analysed in Fig. 7.

5 Future directions and recommendations

While the approaches presented have contributed to shaping and advancing the field of deep MARL, further work is required to optimise the training algorithms and the process of combining the training algorithm with the MARL algorithm, specifically in very large networks such as those that are observed in beyond 5G networks. In addition, the following challenges, currently cited as open research questions, require further investigations.

(1) Learning in real-time, from limited training samples: To adequately train a DNN, the samples of state transitions that have occurred in the operating environment due to the CRs' actions must be collected and fed into the training approach applied. However, collecting these samples from automated systems that

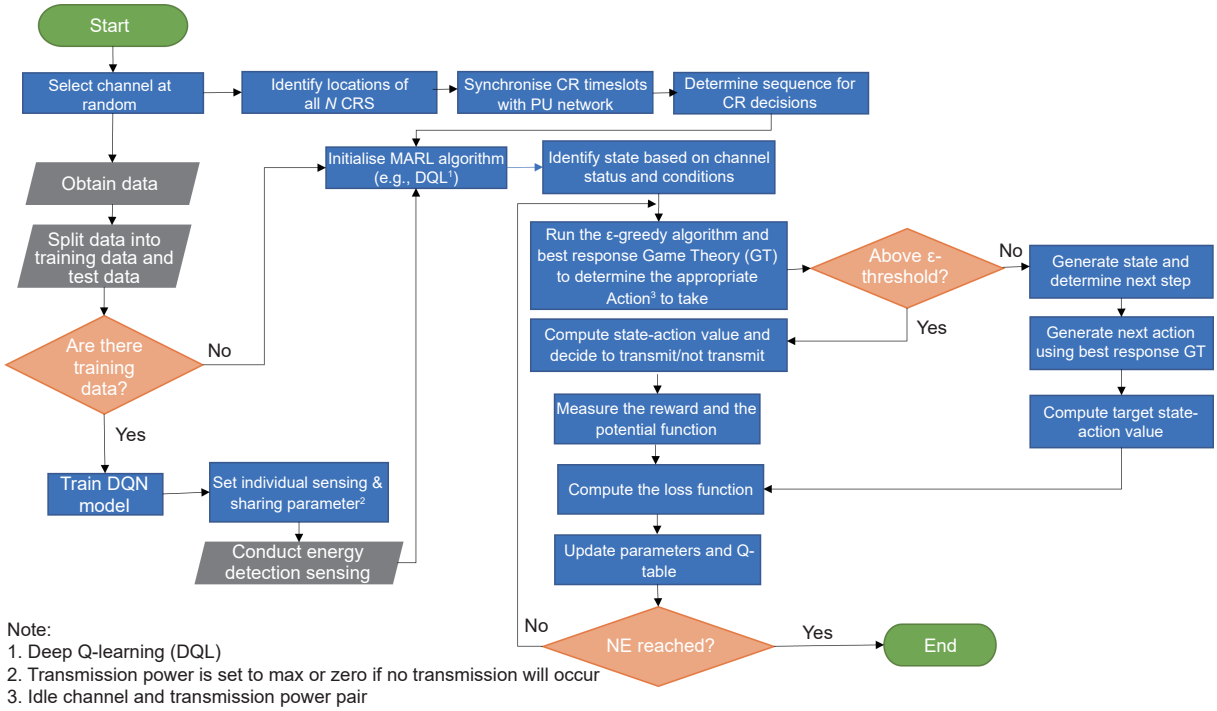

Fig. 7 Potential game based multi-CR learning approach for joint sensing and sharing in CRNs.

interact with the natural world is challenging, time-consuming, and costly. However, without such samples, transfer learning and accurate simulation are not possible. While various research efforts (e.g., Refs. [83, 84]) and workshops (e.g., the workshop described in Ref. [85]) have been held to forge a resolution to the sample efficiency challenge, it remains an open problem.

(2) Sparse or biased rewards: When the feedback the CR receives after taking an action directs the CR in a specific direction or is structured such that the CR does not always get helpful feedback for decisions taken, then learning is prohibited. An example is that a DRL approach produces positive rewards only at certain stages of the game. Such an outcome is deemed to be guided/directed to a target policy instead of the CR having learnt an optimal policy to achieve its objective. Reward shaping, curriculum learning, and curiosity-driven methods are some of the approaches taken to resolve this challenge[56, 86–89]. In addition, Population-Based MARL (PB-MARL) algorithms, a newly established area of MARL, which combine DRL and dynamical population selection methodologies (such as game theory and evolutionary strategies), are deemed useful in producing auto-curricula required for dynamic curriculum learning, thus improving training and rewards feedback. Through this approach, the distributed CR network achieves high parallelism for both data sampling and DNN training. However, none of these approaches has served as a silver bullet approach, making this an open research problem.

(3) Formulating and addressing multi-objective reward functions: The data used for training the DNN may have different effects on the objective that the CR aims to optimise, specifically when the CR intends to optimise multiple objectives, yet is limited by the DRL approach applied to optimise only one objective. Moreover, the policy followed by each player may seem volatile due to the impact of selected actions on the sub-objectives of the reward function. The root cause for this is that training data collected are typically focused on one objective and discount the effect of the selected objects on other underlying objectives. Moreover, current approaches do not facilitate the tracking of multiple objectives as the CR formulates its policy, causing the CR to accept trade-offs the policy is making without knowing what these trade-offs are and without considering alternative decisions. Further details on this challenge are provided in Refs. [90–92].

(4) It was observed that the MARL algorithms applied in CR networks assumed that all the agents had the same learning objective[81]. However, in practice, this assumption is flawed as a distributed CR network can comprise selfish, malicious, and competitive CRs, with opposing learning objectives and thus varying utility functions. Although we have proposed an approach that aligns the objective functions of all the CRs into one potential function, we have not observed other works that have done the same, nor have we tested the proposed approach. Therefore, and as far as we have read, heterogeneous learning remains an open research problem in the field of multi-CR RL.

6 Conclusion

We have presented a comprehensive review of MARL frameworks, methods, techniques, and algorithms and extended this review to the developing field of mean field RL, organic computing, and multi-objective CR RL. We have shown that the CR must demonstrate joint sensing and sharing learning capabilities in a multi-agent environment to become an intelligent wireless communication system capable of unlocking the true power of beyond 5G networks. We also presented an elaborate study of multi-CR RL approaches and proposed a novel approach to address the computational complexity and time duration challenges currently prohibiting RL in CRs. However, given that the CR is resource constrained and CRs in a distributed network may have multiple learning objectives, the joint sensing and sharing multi-CR learning problem remains a complex problem to be solved. To this end, we recommended areas for future work on this topic.

References

[1] E. Biglieri, A. J. Goldsmith, L. J. Greenstein, N. B. Mandayam, and H. V. Poor, *Principles of Cognitive Radio*. New York, NY, USA: Cambridge University Press, 2013.

[2] S. Haykin, Cognitive radio: Brain-empowered wireless communications, *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 2, pp. 201–220, 2005.

[3] K. Rapetswa and L. Cheng, Convergence of mobile broadband and broadcast services: A cognitive radio sensing and sharing perspective, *Intelligent and Converged Networks*, vol. 1, no. 1, pp. 99–114, 2020.

[4] N. Yang, H. Zhang, and R. Berry, Partially observable multi-agent deep reinforcement learning for cognitive resource management, in *Proc. GLOBECOM 2020—2020 IEEE Global Communications Conference*, Taiwan, Province of China, 2020, pp. 1–6.

[5] J. Mitola, Cognitive radio: An integrated agent architecture for software defined radio, PhD dissertation, Teleinfomatics, Royal Institute of Technology (KTH), Stockholm, Sweden, 2000.

[6] C. Clancy, J. Hecker, E. Stuntebeck, and T. O'Shea, Applications of machine learning to cognitive radio networks, *IEEE Wireless Communications*, vol. 14, no. 4, pp. 47–52, 2007.

[7] S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning: From Theory to Algorithms*. New York, NY: Cambridge University Press, 2014.

[8] Y. Lu and K. Yan, Algorithms in multi-agent systems: A holistic perspective from reinforcement learning and game theory, arXiv preprint arXiv: 2001.06487, 2020.

[9] L. Buşoniu, R. Babuška, and B. D. Schutter, Multi-agent reinforcement learning: An overview, in *Innovations in Multi-Agent Systems and Applications – 1*, D. Srinivasan and L. C. Jain, eds. Berlin, Germany: Springer, 2010, pp. 183–221.

[10] N. R. Ravishankar and M. V. Vijayakumar, Reinforcement learning algorithms: Survey and classification, *Indian Journal of Science and Technology*, doi: 10.17485/ijst/2017/v10i1/109385.

[11] E. Tampubolon, H. Ceribasic, and H. Boche, On information asymmetry in competitive multi-agent reinforcement learning: Convergence and optimality, arXiv preprint arXiv: 2010.10901, 2020.

[12] N. Abbas, Y. Nasser, and K. E. Ahmad, Recent advances on artificial intelligence and learning techniques in cognitive radio networks, *EURASIP Journal on Wireless Communications and Networking*, vol. 2015, p. 174, 2015.

[13] M. Bkassiny, Y. Li, and S. K. Jayaweera, A survey on machine-learning techniques in cognitive radios, *IEEE Communications Surveys & Tutorials*, vol. 15, no. 3, pp. 1136–1159, 2012.

[14] M. Gerczuk, Multi-agent reinforcement learning-from game theory to organic computing, https://vixra.org/pdf/1903.0006v1.pdf, 2019.

[15] H. Zhang and T. Yu, Taxonomy of reinforcement learning algorithms, in *Deep Reinforcement Learning*, H. Dong, Z. Ding, and S. Zhang, eds. Singapore: Springer, 2020, pp. 125–133.

[16] K. Zhang, Z. Yang, and T. Başar, Multi-agent

reinforcement learning: A selective overview of theories and algorithms, arXiv preprint arXiv: 1911.10635, 2019.

[17] A. Feriani and E. Hossain, Single and multi-agent deep reinforcement learning for AI-enabled wireless networks: A tutorial, *IEEE Communications Surveys & Tutorials*, vol. 23, no. 2, pp. 1226–1252, 2021.

[18] Y. Yang, Many-agent reinforcement learning, PhD dissertation, Department of Computer Science, University College London (UCL), London, UK, 2021.

[19] A. Alwarafy, M. Abdallah, B. S. Ciftler, A. Al-Fuqaha, and M. Hamdi, Deep reinforcement learning for radio resource allocation and management in next generation heterogeneous wireless networks: A survey, arXiv preprint arXiv: 2106.00574, 2021.

[20] N. C. Luong, D. T. Hoang, S. Gong, D. Niyato, P. Wang, Y. -C. Liang, and D. I. Kim, Applications of deep reinforcement learning in communications and networking: A survey, *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3133–3174, 2019.

[21] A. Harvey, K. B. Laskey, and K. -C. Chang, Machine learning applications for sensor tasking with non-linear filtering, *Sensors*, vol. 22, no. 6, p. 2229, 2022.

[22] G. M. Skaltsis, H. -S. Shin, and A. Tsourdos, A survey of task allocation techniques in MAS, in *Proc. 2021 International Conference on Unmanned Aircraft Systems* (*ICUAS*), Athens, Greece, 2021, pp. 488–497.

[23] M. L. Littman, Markov games as a framework for multi-agent reinforcement learning, in *Proc. Eleventh International Conference*, New Brunswick, NJ, USA, 1994, pp. 157–163.

[24] A. Nowé, P. Vrancx, and Y. -M. D. Hauwere, Game theory and multi-agent reinforcement learning, in *Reinforcement Learning*, M. Wiering and M. Otterlo, eds. Berlin, Germany: Springer, 2012, pp. 441–470.

[25] Y. Yang and J. Wang, An overview of multi-agent reinforcement learning from game theoretical perspective, arXiv preprint arXiv: 2011.00583, 2020.

[26] J. Chen, Q. Yu, P. Cheng, Y. Sun, Y. Fan, and X. Shen, Game theoretical approach for channel allocation in wireless sensor and actuator networks, *IEEE Transactions on Automatic Control*, vol. 56, no. 10, pp. 2332–2344, 2011.

[27] U. Sharma, P. Mittal, and C. K. Nagpal, Implementing game theory in cognitive radio network for channel allocation: An overview, *International Journal of Energy, Information and Communications*, vol. 6, no. 2, pp. 17–22, 2015.

[28] H. -Y. Shi, W. -L. Wang, N. -M. Kwok, and S. -Y. Chen, Game theory for wireless sensor networks: A survey, *Sensors* (*Basel*), vol. 12, no. 7, pp. 9055–9097, 2012.

[29] D. Monderer and L. S. Shapley, Potential games, *Games and Economic Behavior*, vol. 14, no. 1, pp. 124–143, 1996.

[30] M. Bowling and M. Veloso, An analysis of stochastic game theory for multiagent reinforcement learning, https://www.cs.cmu.edu/~mmv/papers/00TR-mike.pdf, 2000.

[31] M. Kearns, M. L. Littman, and S. Singh, Graphical models for game theory, arXiv preprint arXiv: 1301.2281, 2013.

[32] S. Kapoor, Multi-agent reinforcement learning: A report on challenges and approaches, arXiv preprint arXiv: 1807.09427, 2018.

[33] L. P. Kaelbling, M. L. Littman, and A. W. Moore, Reinforcement learning: A survey, *Journal of Artificial Intelligence Research*, vol. 4, pp. 237–285, 1996.

[34] S. Singh, T. Jaakkola, M. L. Littman, and C. Szepesvári, Convergence results for single-step on-policy reinforcement-learning algorithms, *Machine Learning*, vol. 38, no. 3, pp. 287–308, 2000.

[35] J. Hu and M. P. Wellman, Nash Q-learning for general-sum stochastic games, *Journal of Machine Learning Research*, vol. 4, pp. 1039–1069, 2003.

[36] G. Arslan and S. Yüksel, Decentralized Q-learning for stochastic teams and games, *IEEE Transactions on Automatic Control*, vol. 62, no. 4, pp. 1545–1558, 2017.

[37] A. Greenwald, J. Li, and E. Sodomka, Solving for best responses and equilibria in extensive-form games with reinforcement learning methods, in *Rohit Parikh on Logic, Language and Society*, C. Başkent, L. S. Moss, and R. Ramanujam, eds. Cham, Switzerland: Springer, 2017, pp. 185–226.

[38] A. Akramizadeh, M. -B. Menhaj, and A. Afshar, Multiagent reinforcement learning in extensive form games with complete information, in *Proc. 2009 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning*, Nashville, TN, USA, 2009, pp. 205–211.

[39] M. Lanctot, E. Lockhart, J. -B. Lespiau, V. Zambaldi, S. Upadhyay, J. Pérolat, S. Srinivasan, F. Timbers, K. Tuyls, S. Omidshafiei, et al., OpenSpiel: A framework for reinforcement learning in games, arXiv preprint arXiv: 1908.09453, 2019.

[40] C. K. Ling, F. Fang, and J. Z. Kolter, What game are we playing? End-to-end learning in normal and extensive form games, arXiv preprint arXiv: 1805.02777, 2018.

[41] V. S. Borkar, Reinforcement learning in Markovian evolutionary games, *Advances in Complex Systems*, vol. 5, no. 1, pp. 55–72, 2002.

[42] J. Heinrich, M. Lanctot, and D. Silver, Fictitious self-play

in extensive-form games, in *Proc. 32nd International Conference on Machine Learning*, Lille, France, 2015, pp. 805–813.

[43] M. Lanctot, V. Zambaldi, A. Gruslys, A. Lazaridou, K. Tuyls, J. Perolat, D. Silver, and T. Graepel, A unified game-theoretic approach to multiagent reinforcement learning, arXiv preprint arXiv: 1711.00832, 2017.

[44] Y. Wen, H. Chen, Y. Yang, Z. Tian, M. Li, X. Chen, and J. Wang, A game-theoretic approach to multi-agent trust region optimization, arXiv preprint arXiv: 2106.06828, 2021.

[45] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, Playing Atari with deep reinforcement learning, arXiv preprint arXiv: 1312.5602, 2013.

[46] J. N. Tsitsiklis and B. V. Roy, An analysis of temporal-difference learning with function approximation, *IEEE Transactions on Automatic Control*, vol. 42, no. 5, pp. 674–690, 1997.

[47] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al., Human-level control through deep reinforcement learning, *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[48] A. Mosavi, Y. Faghan, P. Ghamisi, P. Duan, S. F. Ardabili, E. Salwana, and S. S. Band, Comprehensive review of deep reinforcement learning methods and applications in economics, *Mathematics*, vol. 8, no. 10, p. 1640, 2020.

[49] P. Hernandez-Leal, B. Kartal, and M. E. Taylor, A survey and critique of multiagent deep reinforcement learning, *Autonomous Agents and Multi-Agent Systems*, vol. 33, no. 6, pp. 750–797, 2019.

[50] P. J. Werbos, *The Roots of Backpropagation: From Ordered Derivatives to Neural Networks and Political Forecasting*. New York, NY, USA: John Wiley & Sons, 1994.

[51] H. Robbins and S. Monro, A stochastic approximation method, *Annals of Mathematical Statistics*, vol. 22, no. 3, pp. 400–407, 1951.

[52] F. P. Such, V. Madhavan, E. Conti, J. Lehman, K. O. Stanley, and J. Clune, Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning, arXiv preprint arXiv: 1712.06567, 2017.

[53] S. Iqbal and F. Sha, Actor-attention-critic for multi-agent reinforcement learning, in *Proc. 36th International Conference on Machine Learning*, Long Beach, CA, USA, 2019, pp. 2961–2970.

[54] X. Ma, Y. Yang, C. Li, Y. Lu, Q. Zhao, and Y. Jun, Modeling the interaction between agents in cooperative multi-agent reinforcement learning, arXiv preprint arXiv: 2102.06042, 2021.

[55] W. Li, B. Jin, X. Wang, J. Yan, and H. Zha, F2A2: Flexible fully-decentralized approximate actor-critic for cooperative multi-agent reinforcement learning, arXiv preprint arXiv: 2004.11145, 2020.

[56] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel, and W. Zaremba, Hindsight experience replay, presented at 31st Conference on Neural Information Processing Systems, Long Beach, CA, USA, 2017.

[57] S. Fujimoto, H. Hoof, and D. Meger, Addressing function approximation error in actor-critic methods, in *Proc. 35th International Conference on Machine Learning*, Stockholm, Sweden, 2018, pp. 1587–1596.

[58] J. Heinrich and D. Silver, Deep reinforcement learning from self-play in imperfect-information games, arXiv preprint arXiv: 1603.01121, 2016.

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

[60] S. Liu, J. Cao, Y. Wang, W. Chen, and Y. Liu, Self-play reinforcement learning with comprehensive critic in computer games, *Neurocomputing*, vol. 449, pp. 207–213, 2021.

[61] A. Nagabandi, G. Kahn, R. S. Fearing, and S. Levine, Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning, in *Proc. 2018 IEEE International Conference on Robotics and Automation* (*ICRA*), Brisbane, Australia, 2018, pp. 7559–7566.

[62] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, Proximal policy optimization algorithms, arXiv preprint arXiv: 1707.06347, 2017.

[63] T. Weber, S. Racanière, D. P. Reichert, L. Buesing, A. Guez, D. J. Rezende, A. P. Badia, O. Vinyals, N. Heess, Y. Li, et al., Imagination-augmented agents for deep reinforcement learning, arXiv preprint arXiv: 1707.06203, 2017.

[64] R. H. Puspita, S. D. A. Shah, G. -M. Lee, B. -H. Roh, J. Oh, and S. Kang, Reinforcement learning based 5G enabled cognitive radio networks, in *Proc. 2019 International Conference on Information and Communication Technology Convergence* (*ICTC*), Jeju, Republic of Korea, 2019, pp. 555–558.

[65] J. Yang, X. Ye, R. Trivedi, H. Xu, and H. Zha, Deep mean

field games for learning optimal behavior policy of large populations, arXiv preprint arXiv: 1711.03156, 2018.

[66] Y. Yang, R. Luo, M. Li, M. Zhou, W. Zhang, and J. Wang, Mean field multi-agent reinforcement learning, in *Proc. 35th International Conference on Machine Learning*, Stockholm, Sweden, 2018, pp. 5571–5580.

[67] J. Subramanian and A. Mahajan, Reinforcement learning in stationary mean-field games, in *Proc. 18th International Conference on Autonomous Agents and MultiAgent Systems*, Montreal, Canada, 2019, pp. 251–259.

[68] M. Agarwal, V. Aggarwal, A. Ghosh, and N. Tiwari, Reinforcement learning for mean field game, arXiv preprint arXiv: 1905.13357, 2019.

[69] A. Angiuli, J. -P. Fouque, and M. Laurière, Unified reinforcement Q-learning for mean field game and control problems, arXiv preprint arXiv: 2006.13912, 2020.

[70] R. Elie, J. Pérolat, M. Laurière, M. Geist, and O. Pietquin, On the convergence of model free learning in mean field games, *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 5, pp. 7143–7150, 2020.

[71] M. Li, Z. Qin, Y. Jiao, Y. Yang, J. Wang, C. Wang, G. Wu, and J. Ye, Efficient ridesharing order dispatching with mean field multi-agent reinforcement learning, in *Proc. WWW '19: The World Wide Web Conference*, San Francisco, CA, USA, 2019, pp. 983–994.

[72] S. G. Subramanian, P. Poupart, M. E. Taylor, and N. Hegde, Multi type mean field reinforcement learning, arXiv preprint arXiv: 2002.02513, 2020.

[73] S. Sudhakara, A. Mahajan, A. Nayyar, and Y. Ouyang, Scalable regret for learning to control network-coupled subsystems with unknown dynamics, arXiv preprint arXiv: 2108.07970, 2021.

[74] M. Muhlhauser, Ubiquitous computing and its influence on MSE [multimedia software engineering], in *Proc. International Symposium on Multimedia Software Engineering*, Taiwan, Province of China, 2002, pp. 48–55.

[75] J. Branke, M. Mnif, C. Müller-Schloer, H. Prothmann, U. Richter, F. Rochner, and H. Schmeck, Organic computing—Addressing complexity by controlled self-organization, in *Proc. Second International Symposium on Leveraging Applications of Formal Methods*, *Verification and Validation* (*isola 2006*), Paphos, Cyprus, 2006, pp. 185–191.

[76] J. Branke, M. Mnif, C. Müller-Schloer, H. Prothmann, U. Richter, F. Rochner, and H. Schmeck, Organic computing—Addressing complexity by controlled self-organization, in *Proc. Second International Symposium on Leveraging Applications of Formal Methods, Verification and Validation*, Paphos, Cyprus, 2006, pp. 185–191.

[77] H. A. Simon, Bounded rationality, in *Utility and probability*, J. Eatwell, M. Milgate, and P. Newman, eds. London, UK: Palgrave Macmillan, 1990, pp. 15–18.

[78] S. Tomforde and B. Sick, eds, *Organic Computing*: *Doctoral Dissertation Colloquium 2018*. Kassel, Germany: Kassel University Press, 2019.

[79] S. Rudolph, S. Tomforde, B. Sick, H. Heck, A. Wacker, and J. Hähner, An online influence detection algorithm for organic computing systems, in *Proc. ARCS 2015—28th International Conference on Architecture of Computing Systems*, Porto, Portugal, 2015, pp. 1–8.

[80] S. Reichhuber and S. Tomlorde, Opportunistic meta-learning: A case study for quality assurance in industry 4.0 environments, in *Proc. 2020 IEEE International Conference on Autonomic Computing and Self-Organizing Systems Companion* (*ACSOS-C*), Washington, DC, USA, 2020, pp. 76–81.

[81] C. Wu, K. Chowdhury, M. D. Felice, and W. Meleis, Spectrum management of cognitive radio using multi-agent reinforcement learning, in *Proc. 9th Int. Conf. Autonomous Agents and Multiagent Systems: Industry Track*, Toronto, Canada, 2010, pp. 1705–1712.

[82] Y. Sun, M. Peng, Y. Zhou, Y. Huang, and S. Mao, Application of machine learning in wireless networks: Key techniques and open issues, *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3072–3108, 2019.

[83] F. E. Dorner, Measuring progress in deep reinforcement learning sample efficiency, arXiv preprint arXiv: 2102.04881, 2021.

[84] A. Kuhnle, M. Aroca-Ouellette, A. Basu, M. Sensoy, J. Reid, and D. Zhang, Reinforcement learning for information retrieval, in *Proc. 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, Virtual Event, Canada, 2021, pp. 2669–2672.

[85] K. Kurzer, P. Schörner, A. Albers, H. Thomsen, K. Daaboul, and J. M. Zöllner, Generalizing decision making for automated driving with an invariant environment representation using deep reinforcement learning, arXiv preprint arXiv: 2102.06765, 2021.

[86] M. Zhou, Z. Wan, H. Wang, M. Wen, R. Wu, Y. Wen, Y. Yang, W. Zhang, and J. Wang, MALib: A parallel framework for population-based multi-agent reinforcement learning, arXiv preprint arXiv: 2106.07551, 2021.

[87] S. Huang and S. Ontañón, Action guidance: Getting the best of sparse rewards and shaped rewards for real-time strategy games, arXiv preprint arXiv: 2010.03956, 2020.

[88] M. Riedmiller, R. Hafner, T. Lampe, M. Neunert, J.

Degrave, T. Wiele, V. Mnih, N. Heess, and J. T. Springenberg, Learning by playing solving sparse reward tasks from scratch, in *Proc. 35th International Conference on Machine Learning*, Stockholm Sweden, 2018, pp. 4344–4353.

[89] G. Dulac-Arnold, D. Mankowitz, and T. Hester, Challenges of real-world reinforcement learning, arXiv preprint arXiv: 1904.12901, 2019.

[90] C. F. Hayes, R. Rădulescu, E. Bargiacchi, J. Källström, M. Macfarlane, M. Reymond, T. Verstraeten, L. M. Zintgraf, R. Dazeley, F. Heintz, et al., A practical guide to multi-objective reinforcement learning and planning, *Autonomous Agents and Multi-Agent Systems*, vol. 36, no. 1, p. 26, 2022.

[91] M. Beeks, R. Refaei Afshar, Y. Zhang, R. Dijkman, C. van Dorst, and S. de Looijer, Deep reinforcement learning for a multi-objective online order batching problem, *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 32, no. 1, pp. 435–443, 2022.

[92] W. Wang, A. Kwasinski, D. Niyato, and Z. Han, A survey on applications of model-free strategy learning in cognitive wireless networks, *IEEE Communications Surveys & Tutorials*, vol. 18, no. 3, pp. 1717–1757, 2016.

Ling Cheng received the BEng degree from Huazhong University of Science and Technology in 1995, and the MSc degree in 2005, and the PhD degree in 2011, both from University of Johannesburg (UJ), South Africa. His research interests are in telecommunications and artificial intelligence. In 2010, he joined University of the Witwatersrand where he was promoted to a full professor in 2019. He serves as an associate editor of three journals. He has published more than one hundred research papers in journals and conference proceedings. He has been a visiting professor at five universities and the principal advisor for over forty full research post-graduate students. He was awarded the Chancellor's medals in 2005 and 2019, and the South Africa National Research Foundation Ratings in 2014 and 2020. The IEEE ISPLC 2015 best student paper award was made to his PhD student in Austin. He is a senior member of IEEE and the vice-chair of IEEE South African Information Theory Chapter.

Kagiso Rapetswa received the MSC (information and electrical engineering) degree from University of Witwatersrand in 2015. She is currently a part-time PhD candidate in University of Witwatersrand. Her research interests include 5G networks, cognitive radios, and machine learning applications in wireless sensor networks.