Application of Reinforcement Learning for Security Enhancement in Cognitive Radio Networks

Mee Hong Ling[a,*], Kok-Lim Alvin Yau[a], Junaid Qadir[b], Geong Sen Poh[c], Qiang Ni[d]

[a]Department of Computing and Information Systems, Sunway University

No. 5 Jalan Universiti, Bandar Sunway, 46150 Petaling Jaya, Selangor, Malaysia

[b]Electrical Engineering Department, School of Electrical Engineering & Computer Science

National University of Science and Technology, Sector H12, Islamabad, Pakistan

[c]University Malaysia of Computer Science & Engineering (UniMy)

Jalan Alamanda 2, Presint 16, 62150 Putrajaya, Malaysia

[d]School of Computing & Communications, Lancaster University

Lancashire LA1 4YW, United Kingdom

*Corresponding author

Phone: +6 03 7491 8622 Ext. 7135

Email: {mhling, koklimy}@sunway.edu.my, junaid.qadir@seecs.edu.pk, poh@unimy.edu.my, q.ni@lancs.ac.uk.

Abstract: Cognitive radio network (CRN) enables unlicensed users (or secondary users, SUs) to sense for and opportunistically operate in underutilized licensed channels, which are owned by the licensed users (or primary users, PUs). Cognitive radio network (CRN) has been regarded as the next-generation wireless network centred on the application of artificial intelligence, which helps the SUs to learn about, as well as to adaptively and dynamically reconfigure its operating parameters, including the sensing and transmission channels, for network performance enhancement. This motivates the use of artificial intelligence to enhance security schemes for CRNs. Provisioning security in CRNs is challenging since existing techniques, such as entity authentication, are not feasible in the dynamic environment that CRN presents since they require pre-registration. In addition these techniques cannot prevent an authenticated node from acting maliciously. In this article, we advocate the use of reinforcement learning (RL) to achieve optimal or near-optimal solutions for security enhancement through the detection of various malicious nodes and their attacks in CRNs. RL, which is an artificial intelligence technique, has the ability to learn new attacks and to detect previously learned ones. RL has been perceived as a promising approach to enhance the overall security aspect of CRNs. RL, which has been applied to address the dynamic aspect of security schemes in other wireless networks, such as wireless sensor networks and wireless mesh networks can be leveraged to design security schemes in CRNs. We believe that these RL solutions will complement and enhance existing security solutions applied to CRN To the best of our knowledge, this is the first survey article that focuses on the use of RL-based techniques for security enhancement in CRNs.

Keywords: Reinforcement learning; trust; reputation; security; cognitive radio networks

1 Introduction

Cognitive radio (CR) [1, 2] is the next-generation wireless communication system that promises to address the artificial spectrum scarcity issue resulting from the traditional static spectrum allocation

policy through dynamic spectrum access. With dynamic spectrum access, unlicensed users, or secondary users (SUs), can opportunistically exploit underutilized spectrum owned by the licensed users or primary users (PUs). Hence, CRs can improve the overall spectrum utilization by improving bandwidth availability at SUs. To achieve these functions, artificial intelligence (AI) techniques have been adopted in CR so that the SUs can sense, learn, and adapt to the dynamic network conditions, in which the PUs' and malicious users' activities appear and reappear. Cognitive radio networks (CRNs) can operate in both centralized and distributed settings: a centralized CRN consists of a SU base station (or access point) that communicates with SU nodes while a distributed network consists of SU nodes that communicate with each other in an ad-hoc manner.

CRNs rely on cooperation for much of their functionality. While such a reliance on cooperative algorithms can make CRNs more efficient, this also opens CRNs up to numerous security vulnerabilities. One of the important requirements of CRNs is that SUs must minimize harmful interference to PUs. This requires SUs to collaborate amongst themselves to perform channel sensing and make accurate final decision on the availability of a channel. However, such collaboration among SUs may pose a security challenge to the SUs' trustworthiness. For instance, in collaborative channel sensing, the legitimate (or honest) SUs depend highly on the dynamic allocation of a common control channel (CCC), which is used for the exchange of control messages during normal operations. However, the collaborating SUs may be malicious, and they may intentionally provide false sensing outcomes to interfere with the PUs or the other SUs, as well as to launch jamming attacks on the CCC, which adversely impacts performance and causes transmissions to come to a halt [3]. Hence, such SUs need to be detected and ignored in collaboration. The aforementioned discussion highlights that CRNs are susceptible to various attacks, such as channel jamming, eavesdropping or packets alteration. Further details on CRNs vulnerabilities, attacks and security threats can be found in detailed survey articles on this topic [4 – 6].

The main security challenge in a CRN is that it operates in a dynamic set of licensed and unlicensed channels (in contrast to traditional wireless networks that typically operate with a fixed set of limited channels). In addition, the dynamic nature of the activities of PUs and malicious nodes requires SUs to change their operating channels from time to time: hence, longer-term knowledge is necessary so

that SUs do not oscillate or constantly switch their actions within a short period of time. With this inherent characteristic, a mechanism to manage and learn from the ever-changing environment is needed to tackle the security challenge.

Reinforcement learning (RL) is an artificial intelligence (AI) approach that helps a decision maker (or agent) to learn the optimal action through repeated interaction with the operating environment [7]. RL is an unsupervised and intelligent approach that enables an agent to observe and learn about the static or dynamic operating environment in the absence of guidance, feedback or the expected response from supervisors (or external critics), and subsequently make decisions on action selection in order to achieve optimal or near-optimal system performance. RL has been adopted in the literature [8 – 16] because it does not require prior knowledge of channel availability and it is highly adaptive to the dynamicity of channels characteristics. In addition, it enables decision makers (or agents) to learn and subsequently achieve near-optimal or optimal solutions in the dynamic environment that may be complex and large-scale in nature [7, 8, 16, 17]. RL has been applied as an alternative to the traditional policy-based approach for system performance enhancement. The policy-based approach requires an agent to follow a set of static and predefined rules on action selection, and it has been traditionally applied to a wide range of application schemes in wireless networks [18].

Several AI approaches, such as RL, artificial neural networks, rule-based system and game-based approaches, have been applied in CRNs to address the security challenge. A comparison of the strengths and limitations of these AI-based security approaches is presented in [19]. In this article, we will focus on RL-based security enhancements in CRNs.

In the following, we will present some advantages of the application of RL to security enhancement in CRNs:

a) RL enables SUs to learn from experience without using an accurate model of the operating environment, and even *without* using any model, allowing the nodes to adapt to their dynamic and uncertain operating environment [20 – 22]. Through adaptation, RL is useful in the identification of SUs' behavior, such as honest SUs that turn malicious [23]. In practice, an issue is that, obtaining the model is a complex

procedure that requires high amount of processing and storage capabilities. The issue of obtaining an appropriate model is faced by many AI schemes, such as game theory (which has been applied to address jamming attacks [24 – 26]) and belief propagation (which has been applied to address primary user emulation attacks [27]). Using RL helps to solve this issue since RL can be applied in a model-free manner.

b) RL enables SUs to make decisions based on a series of actions made, with the notion of maximizing long-term reward, which is more efficient. Using RL helps to solve the performance efficiency issue associated with the game-based approaches, particularly one-shot or repetitive games (e.g., potential game and matrix game), that have been applied to address jamming and primary user emulation attacks [24, 28, 29].

c) RL enables SUs to explore new operating environment and exploit the knowledge gained so far. An issue is that, SUs may converge to a sub-optimal joint action when one of these conditions happens: actions with severe negative rewards exist or multiple high performance actions exist [30]. Using RL helps to solve such issue found in game-based approaches since RL has the flexibility to fine tune its policy as time progresses.

In spite of the advantages being offered by RL, the application of RL to security enhancement has been limited as compared to other application schemes in wireless networks [31] such as routing, scheduling and topology management. Other machine learning algorithms, such as artificial neural networks, support vector machines and K-means have been widely discussed in [32] to tackle CR problems such as malicious SUs reporting inaccurate sensing outcomes in spectrum sensing. While there have been separate surveys on the security aspects of CRNs [4, 6, 17] and the application of RL to CRNs [33], there is lack of a comprehensive survey on the intersection of these two topics. This motivates our study of RL-based security enhancement schemes in CRNs in which we specifically focus on the application of RL as an effective tool to address security issues and provide security enhancement in CRNs. However, due to its novelty, we have also discussed RL models and algorithms applied in wireless

sensor networks (WSNs) and wireless mesh networks (WMNs) in order to leverage to their RL approaches in CRNs.

For clarity, the acronyms used throughout this article are summarized in Table 1; and the generic and specific notations applied in RL models (see Section 4), are presented in Table 2 and Table 3. Note that, the generic notations cover the essential representations of RL, while the specific notations cover the additional representations to improve RL.

The rest of the article is organized as follows. Section 2 presents an overview of RL and its representations. Section 3 presents a taxonomy of RL for security enhancement in CRNs with a minor focus on other kinds of wireless networks, such as WSNs and WMNs. Section 4 presents a survey on the application of RL to security enhancement schemes in CRNs with a minor focus on other kinds of wireless networks, such as WSNs and WMNs. Section 5 provides a discussion on RL performance and complexities. Section 6 presents design consideration for RL models applied to security enhancement schemes. Section 7 discusses open issues. Finally, we provide conclusions in Section 8.

Table 1 Acronyms used in this article.

Acronym	Description
AI	Artificial intelligence
CCC	Common control channel
CR	Cognitive radio
CRN	Cognitive radio network
MARL	Multi-agent reinforcement learning
PHC	Policy hill-climbing
PU	Primary user
RL	Reinforcement learning
SU	Secondary user
TRM	Trust and reputation management
WMN	Wireless mesh network
WoLF	Win-or-learn-fast
WSN	Wireless sensor network

Table 2 Generic notations for RL models and algorithms.

	Notation	Description	Example representations	Reference	Section
General representation for RL	τ	One time unit	One time unit can be either a time slot Δt, a time window t or an episode e_τ		
	ε	Exploration probability			
	α	Learning rate			
	γ	Discount factor			
	$s^i_{j\in J,\tau} \in S$	State j observed by node i at time τ	State set S represents four sub-states including the presence of PU in a spectrum, node i's gain, the number of jammed control channels in the spectrum, and the number of jammed data channels in the spectrum	Wang et al. [39]	4.2
			State set S represents the channel conditions whether PU occupies the channel, a malicious SU jams the channel or there is successful transmission in the channel	Wu et al. [48]	4.3
			State set S represents whether channel is not occupied or not occupied by a PU	Lo et al. [49]	4.4
			State set S represents the reputation values of a neighbor node	Maneenil et al. [36]	4.5.2
	$a^i_{k\in K,\tau} \in A$	Action k taken by node i at time τ	Action set A represents the choice of potential SU collaborator nodes	Vučević et al. [38]	4.1
			Action set A represents the choice of transmissions in control and data channels	Wang et al. [39]	4.2
			Action set A represents the choice of staying or switching a channel	Wu et al. [48]	4.3
			Action set A represents the number of CCCs selected for transmission	Lo et al. [49]	4.4
			Action set A represents the choice of upstream nodes	Mistry et al. [43]	4.5.1(a)
			Action set A represents the choice of potential collaborator nodes	Mukherjee et al. [53]	4.5.1(b)
			Action set A represents the choice of potential neighbor nodes to forward packets to destination	Maneenil et al. [36]	4.5.2
	$r^i_{m\in M,\tau+1}(s^i_{j\in J,\tau}, a^i_{k\in K,\tau}) \in R$	Delayed reward m received by node i at time $\tau+1$ after taking action k in state j at time τ	Reward set R represents the number of false alarms made by a SU collaborator node within a time window	Vučević et al. [38]	4.1
			Reward set R represents the spectrum gain when the selected channel is unjammed	Wang et al. [39]	4.2
			Reward set R represents the channel gain for selecting an unjammed channel	Wu et al. [48]	4.3
			Reward set R represents the number of valid CCCs that are not occupied by PUs, jammed-free and common to SUs selected for transmission	Lo et al. [49]	4.4
			Reward set R represents the reputation value of an upstream node	Mistry et al. [43]	4.5.1(a)
			Reward set R represents the reputation value of potential collaborator nodes	Mukherjee et al. [43]	4.5.1(b)
			Reward set R represents a constant value to be rewarded to all nodes within a route after an episode	Maneenil et al. [36]	4.5.2
	$Q^i_\tau(s^i_{j\in J,\tau}, a^i_{k\in K,\tau})$	Q-value of action k in state j at time τ	Learnt action value that is updated using previous Q-value, delayed reward and discounted reward	Equation (1)	2.1
	$\pi^{i,*}(s^i_{j\in J,\tau})$	Optimal policy for node i in state j at time τ			
	$V^{\pi^i}(s^i_{j\in J,\tau})$	State value for node i in state j operated under node i's policy at time τ			

Table 3 Specific notations for RL models and algorithms.

Model	Notation	Description
RL with suitability value [38]	$N_{k,\tau}^{i}$	Number of false alarms made by node i's collaborator node k within a time window $\tau = t$
	$\pi_{k,\tau}^{i}$	The probability that node i chooses node k at a time window $\tau = t$
RL with Minimax Q-learning [39]	$P_{j,\tau}^{i}$	The presence of PU in spectrum j of node i at time slot $\tau = \Delta t$
	$g_{j,\tau}^{i}$	The gain of PU in spectrum j received by node i at time slot $\tau = \Delta t$
	$H_{j,C,\tau}^{i}$	The number of jammed control channels in spectrum j observed by node i at time slot $\tau = \Delta t$
	$H_{j,D,\tau}^{i}$	The number of jammed data channels in spectrum j observed by node i at time slot $\tau = \Delta t$
	$\pi^{*}(s_{j,\tau}^{i})$	The optimal policy for node i in spectrum j at time slot $\tau = \Delta t$
	$V(s_{j,\tau}^{i})$	The state value node i in spectrum j at time slot $\tau = \Delta t$
	α_{τ}	Learning rate at time slot $\tau = \Delta t$
RL with decreasing rate [48]	R	The successful transmission gain
	C	The cost of switching to unutilized or unjammed channel
	L	The cost of switching to a jammed channel
RL with PHC and WoLF [49]	$U_{j=0,k,\tau}^{i}$	The number of valid CCCs which are unoccupied by PUs, jammed-free and common to SUs selected for packet transmission
	δ	The step size for policy update
RL with discount factor $\gamma = 0$ [43,53]	Q_{th}	Q-value threshold
	$p_{k,\tau}^{i}$	The number of accurate final decisions at time window $\tau = t$ when node k is chosen by node i
	$q_{k,\tau}^{i}$	The number of inaccurate final decisions at time window $\tau = t$ when node k is chosen by node i
	$\varepsilon_{k,\tau}^{i}$	The relative error at time window $\tau = t$ when node k is chosen by node i
	μ_{i}	The mean of the predicted errors of node i
	σ_{i}	The standard deviation of the predicted errors of node i

2 Reinforcement Learning

This section presents the generic model and algorithm, as well as the flowchart, of a popular RL approach, namely Q-learning. This section serves as a foundation for more advanced RL models and algorithms applied to security enhancement schemes presented in Section 4.

Q-learning is an on-line algorithm in RL [7, 34]. On-line learning enables an agent to learn in an interactive manner with the operating environment as the agent operates. Figure 1 shows an abstract view of RL, which can embedded in an agent (or a SU). RL consists of three main elements, namely *state*, *action* and *reward.* The *state*, which is observed from the operating environment, represents the factors that affect the way in which an agent makes a decision (e.g., the availability of a spectrum in an anti-jamming scheme). The state may not be represented in some RL models as the changes to the environment do not affect an agent's action selection, and hence, such model is called *stateless*. The *reward* represents the performance metrics to be maximized (e.g., detection rate), or minimized (e.g., probabilities of false positive and false negative). The *action* represents an action taken by an agent in order to maximize its reward (e.g., the choice of control and data channels to transmit messages in an anti-jamming scheme). For instance, with respect to a security enhancement scheme that applies reputation, state $s^i_{j\in K,\tau} \in S$ represents the reputation value of a neighbor node $j \in J$, where J represents the set of neighbor nodes of node i [35, 36], action $a^i_{k\in K,t} \in A$ represents the selection of a neighbor node k by node i to forward packets towards destination [35, 36], and reward $r^i_{m\in M,\tau}\left(a^i_{k\in K,\tau}\right) = +1$ represents a constant value to be rewarded to all nodes in a route after an episode e_τ, which is the time required to establish a route. This means that a selected node for a route will only receive its reward after a route has been established [35, 36]. Note that, the reward may be received at every episode $e_{\tau+1}$, which is comprised of a number of time instants, or at every time instant τ . As an example for the latter case, a node i chooses sub-channel k at time τ and receives reward $r^i_{k,\tau+1}\left(a^i_\tau\right) = (\mathrm{S/N})$, which represents a signal-to-noise-ratio (SNR) value, at time $\tau + 1$ [37]. In the rest of this subsection, to enhance the readability, the notations are simplified by considering a single SU in the operating environment. For instance, we write state s_τ to represent state $s^i_{j\in J,\tau}$ although both refer to the state.

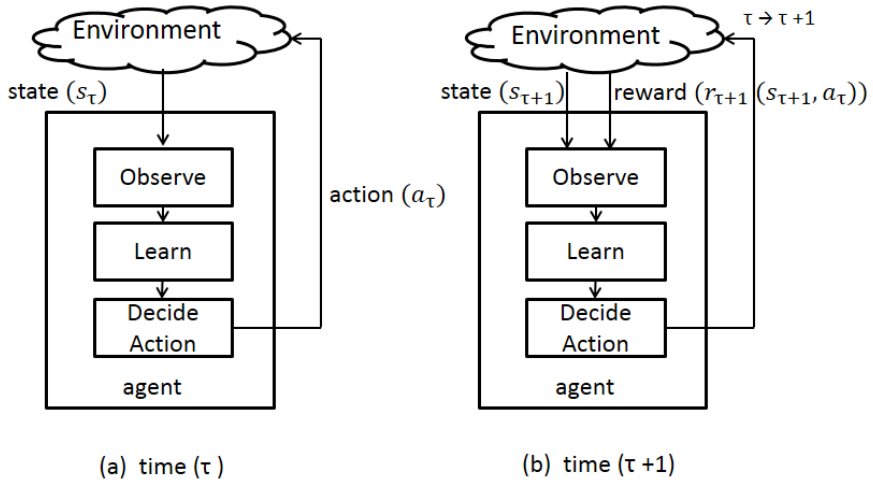

Figure 1: Abstract view of a RL agent in its operating environment for each learning cycle.

Q-learning estimates the Q-values of state-action pairs $Q_\tau(s_\tau, a_\tau)$. For each state-action pair, an agent observes its short-term reward $r_{\tau+1}(s_{\tau+1}, a_\tau)$, and learns its future reward as time progresses. The short-term reward $r_{\tau+1}(s_{\tau+1}, a_\tau)$ is received at time $\tau + 1$ after an agent has taken the action a_τ at time τ. The future reward $\gamma \max_{a \in A} Q_\tau(s_{\tau+1}, a)$ represents the cumulative rewards received by an agent at time $\tau + 1, \tau + 2, \cdots$ [7]. The short-term reward is called the "*delayed reward"* in RL terminology while the future reward is called the "*discounted reward"*. The Q-value $Q_\tau(s_\tau, a_\tau)$ is updated as follows:

$$Q_{\tau+1}(s_\tau, a_\tau) \leftarrow (1 - \alpha) Q_\tau(s_\tau, a_\tau) + \alpha [r_{\tau+1}(s_{\tau+1}, a_\tau) + \gamma \max_{a \in A} Q_\tau(s_{\tau+1}, a)] \quad (1)$$

where the *learning rate* $0 \leq \alpha \leq 1$ determines the extent to which the newly acquired knowledge overrides the previously learnt Q-value $Q_\tau(s_\tau, a_\tau)$; and the *discount factor* $0 \leq \gamma \leq 1$ emphasizes on the importance of future rewards. The higher the learning rate α, the greater the current learnt Q-value overrides its old value, and this speeds up the learning process which may lead to faster convergence; however, this may destabilize the learning process causing the agent fail to converge. On the other hand, the lower the learning rate α, the smoother the learning process albeit at the potential cost of convergence taking longer. If $\alpha = 1$, an agent considers the most current Q-value only. The higher the discount factor γ, the greater the agent relies on the discounted reward, which is the maximum Q-value of the next state-

action pair. If $\gamma = 1$, an agent considers the same weightage for both delayed and discounted rewards. If $\gamma = 0$, the agent only considers maximizing the short-term delayed reward, and in this case, it is called a myopic approach.

The state-action pairs and their respective delayed and discounted rewards are represented by Q-values, which are kept in a two-dimensional $|S| \times |A|$ Q-table. The action taken is based either on an optimal policy π^* (using "*exploitation*" in the RL terminology) or a random policy (called "*exploration*"). When an agent selects an appropriate action for a particular state, the agent receives positive delayed reward, and so the respective Q-value increases, and vice-versa. Hence, in order to maximize the cumulative reward $V^{\pi^*}(s_\tau)$ (or the value function) over a period of time, an agent learns to take the optimal or near-optimal policy π^* (or a series of actions), which has the optimal Q-value given a particular state as follows:

$$V^{\pi^*}(s_\tau) = \max_{a \in A} Q_\tau(s_\tau, a) \tag{2}$$

Hence, the optimal policy is

$$\pi^*(s_\tau) = \operatorname{argmax}_{a \in A} Q_\tau(s_\tau, a) \tag{3}$$

Both exploration and exploitation complement each other. To *explore,* an agent takes random actions. While this may result in lesser accumulation of reward in the short term, it may lead to the discovery of an action that yields better accumulated rewards in the future. There are two popular exploration policies, namely ε*-greedy* and *softmax* action selection approaches. The ε-greedy policy chooses exploration actions with a small probability ε and exploitation actions with probability $1 - \varepsilon$. The softmax policy, on the other hand, chooses exploitation actions with probability $e^{Q_t(a)/T} / \sum_{b=1}^{|A|} e^{Q_t(b)/T}$, where $Q_\tau(s, a)$ represents the knowledge (or Q-value) of state s and action a at time τ, and T represents temperature. Higher T increases exploration rate while lower T increases

exploitation rate, and so the temperature is reduced as time goes by. To *exploit*, an agent chooses an action that has the highest Q-value using Equation (3) to provide the highest possible accumulated reward. To achieve an optimal or near-optimal performance, a balanced trade-off between exploration and exploitation is required. Hence, RL is suitable for most problems that require an agent to learn and re-learn from an uncertain or changing operating environment in order to achieve a given goal.

As seen from Algorithm 1 and Figure 1(a), at time τ an agent observes the current state s_τ, chooses and executes an action a_τ for the current state s_τ. We consider the ε-greedy approach in this case. At time $\tau + 1$, an agent receives delayed reward $r_{\tau+1}(s_{\tau+1}, a_\tau)$, which is the consequence of its action a_τ at time τ (see Fig 1(b)). The agent also observes the state $s_{\tau+1}$ to identify its discounted reward $\gamma \max_{a \in A} Q_\tau(s_{\tau+1}, a)$. The Q-value $Q_\tau(s_\tau, a_\tau)$ for state s_τ and action a_τ is updated using the delayed and discounted rewards $r_{\tau+1}(s_{\tau+1})$ is updated using Equation (1). The iteration repeats until the Q-values converge.

Algorithm 1 Q-Learning algorithm.

Repeat

a) Observe current state s_τ

   Determine exploration or exploitation

   i. If exploration, choose a random action

   ii. If exploitation, choose the best known action using Equation (3)

b) Choose action a_τ

c) Receive $r_{\tau+1}(s_{\tau+1})$

d) For state-action pair (s_τ, a_τ), update $Q_\tau(s_\tau, a_\tau)$ using Equation (1)

Figure 2 represents Algorithm 1 as a flowchart with the respective step in the algorithm shown in the figure. The flowchart has been prepared for ease of comparison with the flowcharts for various RL algorithms presented in Section 4.

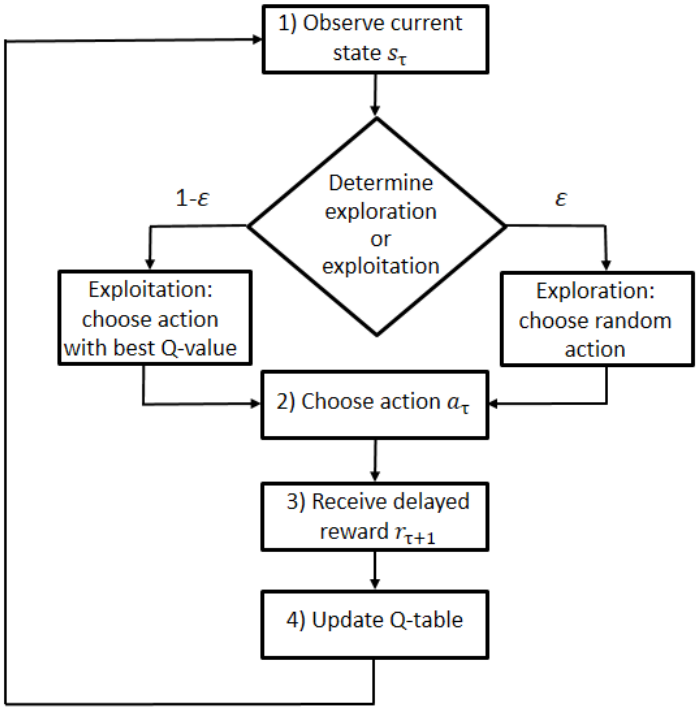

Figure 2: Q-learning flowchart.

3. Security Enhancement Taxonomy in Cognitive Radio Networks

In this section, we discuss the security vulnerabilities associated with CRNs and application schemes, with a minor focus on WSNs and WMNs. Our discussion covers the application and security enhancement schemes, the types of attacks as well as the challenges, characteristics, and performance metrics associated with the different security enhancement schemes. To provide further insights on this topic, and to make our article comprehensive in order to leverage on additional RL schemes, we also discuss RL models and algorithms applied to security enhancement schemes in WSNs and WMNs. A taxonomy of RL for wireless security enhancement is shown in Figure 3, and it is discussed in the rest of this section.

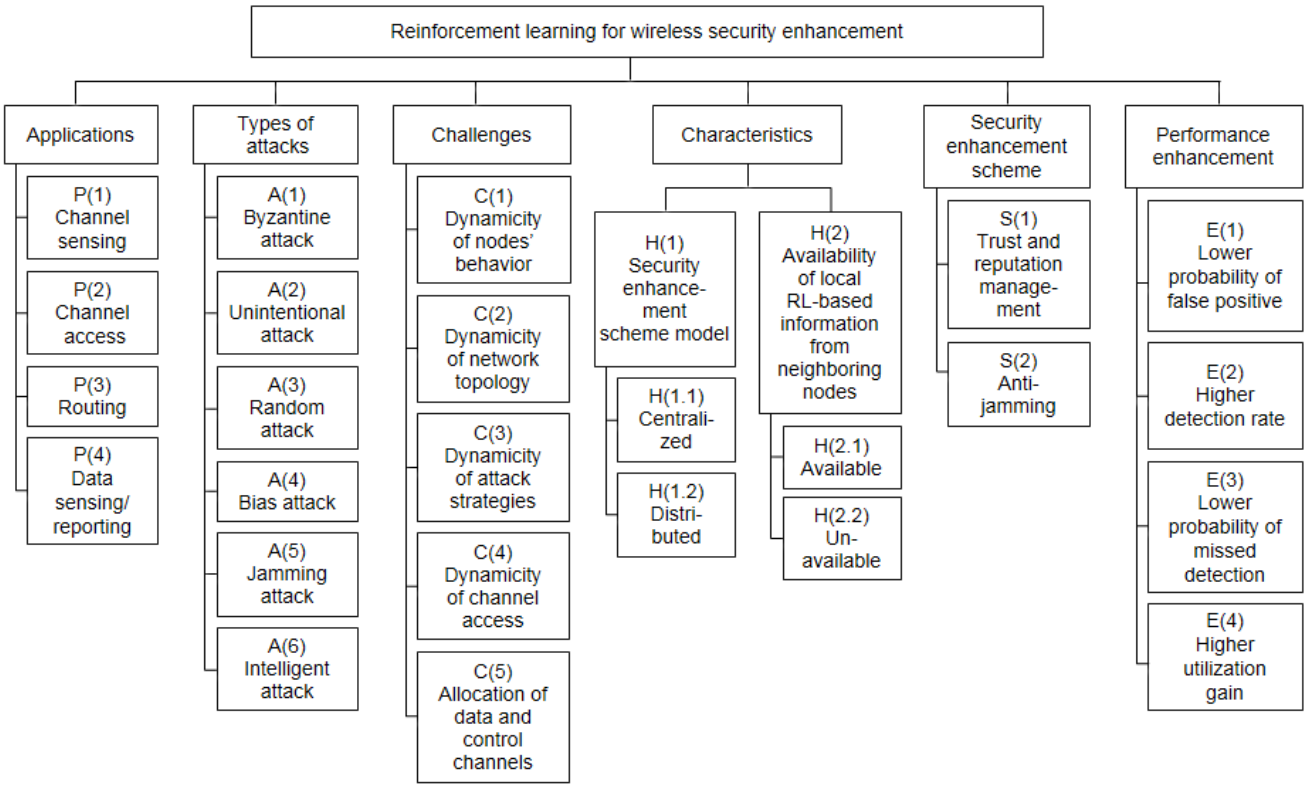

Figure 3: Taxonomy of RL in wireless networks for security enhancement

3.1 Types of wireless networks and application schemes

Generally speaking, there are two types of application schemes with security vulnerabilities in which RL has been applied to address in CRNs as follows:

P.1 *Channel sensing*. CR uses channel sensing technique, such as energy detection, to sense for white spaces. Generally speaking, exploration enables SUs to detect the white spaces at the earliest possible time, while exploitation enables SUs to efficiently access and utilize the channel(s) [38].

P.2 *Channel access*. CR enables SUs to opportunistically access underutilized channels without causing unacceptable interference to the PUs' activities. While this mechanism enables SUs to maximize the usage of the underutilized channels, it poses some security vulnerabilities whereby the malicious SUs can do likewise with the intention to jam the channels so as to deprive honest SUs from accessing the channels [39].

3.1.1. Other wireless networks

In this subsection, we describe RL-based applications in WSNs and WMNs in which the RL models and algorithms applied to these networks can be leveraged to design security schemes in CRNs.

3.1.1.1 Wireless mesh networks

Wireless mesh network (WMN) is a wireless network where each node can communicate directly with one or more neighbor nodes in the absence of fixed network infrastructure [40]. The nodes are made up of mesh routers and mesh clients that help to forward packets in multiple hops. A WMN is self-organized and self-configured in nature, in which the nodes can automatically establish and maintain connectivity among themselves throughout a distributed and dynamic network. RL has been applied to address the security vulnerabilities for the routing application in WMNs as explained below:

P.3 *Routing.* In WMNs, nodes rely on forwarding by their neighboring nodes for packet delivery through multi-hop routing/ forwarding. A WMN is vulnerable to attacks as some nodes in the route may be malicious. As such, it is vital to ensure that nodes in a route are non-malicious. In previous work, RL has been applied to select a next-hop neighbor node [36].

3.1.1.2 Wireless sensor networks

Wireless sensor network (WSN) comprises autonomous sensor nodes that collaboratively monitor the surrounding environment. The nodes send their sensing outcomes to a sink node that performs data fusion and make final decision on sensing outcomes. Sensor nodes are mostly battery powered and energy constrained. In addition, the limited computational capability and memory of sensor nodes, along with its energy-constrained nature, means that there is a high probability of node failure [41, 42]. Thus, even though the sensor nodes are mostly stationary, the topology of WSNs can be dynamic due to node failure. In WSNs, RL has been applied to address the following application scheme with potential security vulnerabilities:

P.4 *Data sensing/reporting*. In WSNs, a sensor node must ensure the accuracy of the final decision on sensing outcomes. This requires both the sensor nodes and their respective upstream sensor nodes leading to sink nodes to be trustworthy. In previous work, RL has been applied to select honest upstream nodes [43].

3.2 Types of attacks

There are *six* distinct types of attacks that RL has been applied to address:

A.1 *Byzantine attack*. In this attack, malicious nodes appear to be honest nodes [23] and use their privilege to disrupt the communication of other nodes in the network without consideration of its own resource consumption [44]. As the malicious nodes appear to be honest, it can generate some mistrust amongst honest nodes.

A.2 *Unintentional attack.* In this attack, non-malicious nodes unintentionally launch attacks (e.g., generating inaccurate sensing outcomes and making incorrect decisions) due to manipulation from malicious nodes, technological limitation (e.g., hardware and software errors), or due to environmental factors (e.g., sensors located in shadowing zone) [38]. As these attacks are unintentional, some measures need to be incorporated in the detection mechanism to ensure that these non-malicious nodes are not misdetected as malicious [45].

A.3 *Random attack.* In this attack, malicious nodes launch attacks at random. For instance, malicious nodes introduce errors in the sensing outcomes [43], or jam the channels randomly [39]. Such attacks affect the honest nodes so that they are unable to predict the malicious nodes' next course of actions. As these attacks are hard to predict, the honest nodes would need to learn and re-learn the next possible course of actions to be taken by the malicious nodes.

A.4 *Bias attack.* In this attack, malicious nodes launch attacks systematically and intentionally in a collaborative manner. For instance, the malicious nodes may either send inaccurate sensing outcomes [43], or jam the channels in order to reduce channel access opportunities of honest nodes [39]. As these attacks are biased, the damaging

effects to honest nodes may be higher than that of other attacks, such as random attacks.

A.5 *Jamming attack.* In this attack, malicious nodes keep the network busy by constantly sending packets on a particular channel, or intentionally cause high interference to disrupt the network in order to reduce the signal-to-noise ratio significantly, and subsequently prevent efficient utilization of the channels [36]. A malicious SU may choose to launch an attack on the CCC to inflict maximum damage since many important operations, such as the exchange of control messages, take place on the CCC. These control messages may include sensing outcomes, routing information and coordination information of channel access [46].

A.6 *Intelligent attack.* In this attack, malicious nodes maximize their attack performance by leveraging on AI techniques. For instance, in Sybil attack [45], which enables nodes to have multiple identities, the malicious nodes make use of RL to learn the optimal number of false sensing outcomes to subvert collaborative spectrum sensing. This is to prevent their malicious intentions being detected [47].

3.3 Challenges

There are *five* challenges associated with security enhancement schemes as follows:

C.1 *Dynamicity of nodes' behavior*. Nodes may change from honest to malicious as time progresses, and vice-versa [38]. RL has been applied to provide a dynamic and intelligent mechanism to monitor the nodes' behavior at all times.

C.2 *Dynamicity of network topology*. Nodes may be added or removed from wireless networks in the absence of a centralized base station. With such flexibility, it opens up security vulnerabilities as the malicious nodes could choose to leave the network either before they have been detected as malicious or otherwise [43]. RL has been applied to provide a network with honest nodes for collaboration, and hence, it improves the efficiency of the network.

C.3 *Dynamicity of attack strategies*. Malicious nodes' strategies may change dynamically and it is a challenge to keep track of the way (e.g., frequency) they attack [39]. RL has been applied to provide a dynamic and intelligent mechanism to learn malicious nodes' strategies as time progresses.

C.4 *Dynamicity of channel access*. During a channel switch, a SU chooses the next operating channel that provides at least, if not better, channel quality and bandwidth leading to network performance enhancement. However, malicious SUs may launch jamming attacks A(5) on those better channels, which are likely to be chosen, making the channel switch ineffective. RL has been applied to provide a better channel transition [48].

C.5 *Allocation of data and control channels.* Optimal allocation of data and control channels maximizes CRN performance. The allocation is vulnerable to attacks as malicious nodes may choose to jam the control channel to achieve maximum effects. RL has been applied to learn the strategy for optimal channel allocation among the honest SUs [49].

3.4 System characteristics

The *two* characteristics associated with security enhancement schemes are as follows:

H.1 *Security enhancement scheme model: centralized* H(1.1) or *distributed* H(1.2). There are two types of models, namely centralized and distributed models. Centralized model is normally embedded in a centralized entity, such as a base station or a decision fusion node in centralized networks; while distributed models are normally embedded in distributed entities, such as the hosts in distributed networks. In centralized model, a central node, such as a base station and a fusion center, may have the knowledge, such as sensing outcomes, of most of its hosts; while in distributed model, every node may have the knowledge of its neighbor nodes only.

H.2 *Availability of local RL-based information from neighboring nodes: available* H(2.1) or *unavailable* H(2.2). The local RL-based information (i.e., state, selected action and

received reward) from neighboring nodes may be available or unavailable to a node. The availability H(2.1) of the information enables nodes to make decisions on action selection without jeopardizing network performance of neighboring nodes; however, message exchanges may be necessary, and so it incurs control overhead. The unavailability H(2.2) of the information requires nodes to make decisions on action selection independently.

3.5 Existing schemes

The RL approach has been applied in *two* security enhancement schemes as follows:

S.1 *Trust and reputation management (TRM)*. Cooperation enables nodes to achieve, through collective efforts, a common or network-wide objective. During collaboration, the collaborating nodes exchange information and use the collective information to make final decisions on the action selection. While authentication, authorization and access control may detect the malicious nodes or manipulated information, attackers may still exist and manipulate the decision [45, 50]. Hence, it is necessary to detect malicious nodes and anomalous events, which deviate from an expected range of system behavior. In general, a threshold characterizes the expected range of system behavior. For instance, the operating parameter should be smaller than the threshold. TRM calculates the reputation value of each node, and subsequently uses the value to identify malicious users among its collaborating nodes.

Higher reputation values indicate higher degree of legitimacy (or smaller deviations from the thresholds). RL has been applied in TRM to identify malicious nodes and address the challenge of dynamicity in the malicious nodes' behavior C(1) [38].

S.2 *Anti-jamming*. Due to the intrinsic nature of CRN that allows SUs to switch from one channel to another in order to maximize the use of the underutilized channels, this characteristic has opened up some security vulnerabilities. The malicious SUs may intentionally jam the white spaces by occupying them through constant transmission of

signals in the white spaces [51] or by producing high interference to starve honest SUs and prevent them from transmitting [52]. Although malicious nodes may target other channels, an intelligent malicious SU node may prefer to jam control channels as it can cause denial of service in the networks. RL has been shown to be effective in tackling jamming [39] due to its ability to learn from the operating environment and adapt quickly to changes.

3.6 Performance enhancements

The RL approach has been shown to achieve the following *four* performance enhancements.

E.1 *Lower probability of false positive.* False positive/ alarm is triggered when honest nodes and their respective activities are incorrectly identified as malicious in nature. RL can help reduce the false positives/ alarms [38, 53].

E.2 *Higher detection rate.* Detection rate is the accuracy of identifying malicious nodes. It can be measured in terms of the number of iterations or cycles required to detect the malicious nodes. RL can help improve the detection rate [43, 53].

E.3 *Lower probability of missed detection.* Missed detection happens when malicious nodes and their respective activities are incorrectly identified as honest nodes. With RL, the probability of missed detection is reduced [38, 53].

E.4 *Higher utilization gain.* Utilization gain can be measured by network performance in terms of data throughput, packet loss or delay. Higher utilization gain indicates higher data throughput, lower packet loss and lower delay. Lower utilization gain may be triggered by attacks, such as jamming activities caused by malicious SUs. RL can help improve the utilization gain [39, 48, 49].

3.7 The role of RL in wireless security enhancement

To further expound on Figure 3, we present the insights on the role of RL in the aspects of applications, types of attacks, challenges, characteristics and security enhancement scheme in Table 4.

Table 4 The role of RL in wireless security enhancement.

Cat-ego-ries	Types	Description	
Applications	P(1) Channel sensing	RL helps honest SUs to detect white space accurately.	
	P(2) Channel access	RL maximizes the white spaces usage among the honest SUs without causing unacceptable interference to PUs.	
	P(3) Routing	RL ignores malicious nodes along a route in the network.	
	P(4) Data sensing/ reporting	RL ensures the accuracy of the sensing outcomes.	
Types of attacks	A(1) Byzantine attack	RL monitors the dynamicity of SUs' behaviour so that it can detect SUs who turn malicious as time progresses.	
	A(2) Unintentional attack	RL enables honest SUs to learn and identify non-malicious SUs with malfunction features so as not to deprive them from collaborating with other SUs.	
	A(3) Random attack	RL enables honest SUs to learn the malicious SUs' attack strategies in order to predict their next course of action, which is random in nature.	
	A(4) Bias attack	RL enables honest SUs to identify the pattern of the attacks, which is collaborative in nature.	
	A(5) Jamming attack	RL enables honest SUs to find optimal control and data channel allocation strategies in order to maximize channel utilization in the presence of jamming from malicious nodes.	
	A(6) Intelligent attack	RL enables honest SUs to counter malicious SUs who leverage on learning and intelligent mechanisms, such as learning the optimal number of false sensing outcomes to be sent to the decision fusion.	
Challenges	C(1) Dynamicity of nodes' behavior	RL learns a nodes' behavior as time progresses.	
	C(2) Dynamicity of network topology	RL learns and identifies honest SUs in a dynamic network environment for collaboration purpose.	
	C(3) Dynamicity of attack strategies	RL learns the malicious nodes' strategies as time progresses.	
	C(4) Dynamicity of channel access	RL learns and identifies a high-quality channel during a channel switch.	
	C(5) Allocation of data and control channels	RL learns the strategy for an optimal channel allocation among the honest SUs.	
Characteristics	H(1) Security enhancement scheme model	H(1.1) Centralized	RL keeps learnt knowledge of all SUs in a fusion center or base station decision making.
		H(1.2) Distributed	RL keeps learnt knowledge of neighboring SUs in every SU for decision making.
	H(2) Availability of local RL-based information	H(2.1) Available	The local RL-based information from neighbor nodes is available to SU to make decision on action selection.
		H(2.2) Unavailable	The local RL-based information from neighbor nodes is not available to SU, and hence, it needs to make action selection independently.
Scheme	S(1) TRM scheme	RL is applied to TRM scheme to learn the reputation values of the SUs so as to detect malicious SUs.	
	S(2) Anti-jamming	RL is applied to anti-jamming scheme to learn jamming from malicious SUs so as to avoid those channels.	

4 Application of Reinforcement Learning for Security Enhancement in Cognitive Radio Networks

This section presents RL models with respect to their applications, algorithms used, types of challenges and attacks for CRNs (see Sections 4.1 – 4.4). As a comprehensive survey article, we have included RL-

based applications in WSNs (see Section 4.5.1) and WMNs (see Section 4.5.2), which are limited in the literature, in order to provide further insight on the leveraging of such RL models and algorithms to CRNs. To do this, we describe how to address the core challenges of CRNs (such as channel dynamicity) while providing support to other minor challenges in CRNs, such as energy conservation while leveraging the RL models to CRNs. Design considerations and a guideline for the application of RL to CRN is provided in Section 6.1. Table 5 summarizes the RL models, which have been applied to security schemes in the literature. Table 5 also compares various aspects of RL models in the aspects of strength, attacks and challenges. The complexity of various RL algorithms has also been incorporated into the respective tables (Algorithms 2–7). For consistency purpose, we standardize the various notations used in the respective articles as shown in Table 2 and Table 3.

Table 5 Description of RL model and its strength to handle attacks in the midst of challenges.

Reinforce-ment learning model	Description	Strength	Attack types					Challenges				
			A.1 Byzantine attack	A.2 Unintentional attack	A.3 Random attack	A.4 Bias attack	A.5 Jamming attack	C.1 Dynamicity of nodes' behavior	C.2 Dynamicity of network topology	C.3 Dynamicity of attack strategies	C.4 Dynamicity of channel access	C.5 Allocation of data and control channels
RL with suitability value [38]	Each SU calculates a suitability value for each neighbor using softmax, and uses it to update the Q-value of the neighbor, in order to select honest neighbors for collaboration.	The suitability value helps to calculate the Q-value of a state-action pair of each neighbor node accurately even though there may be lower number of honest neighbors, which may vary significantly.	√	√				√				
RL with minimax Q-learning [39]	Each SU learns attack strategies from malicious SUs, and selects control or data channels, in order to avoid jammed channels.	Minimax Q-learning helps SUs to learn the dynamic attack strategies from malicious SUs with reduced learning activities.					√			√		
RL with decreasing learning rate [48]	Each SU updates the learning rate as time progresses and selects the next operating channel that is less likely to be jammed.	The traditional RL approach is used with reducing learning rate while ensuring convergence in an operational cycle.					√			√	√	
RL with PHC and WoLF [49]	Each SU updates policy in adaptation to the operating environment. For example, when the reward value is greater than its threshold (e.g., when the PU's activity is low), an agent increases the policy step size, which is used to update the policy. This ensures faster learning =to avoid attacks by malicious SUs.	The multi-agent RL model has the ability to update policies using different step sizes in accordance to the operating environment. So, it learns faster whenever the operating environment is favorable. This increases the convergence rate to the optimal policy.					√					√
RL with discount factor $\gamma = 0$ [43]	Each node updates Q-value with delayed reward only with discount factor $\gamma = 0$ in order to select a node for collaboration.	This model does not depend on future rewards that may be highly dynamic.			√	√		√				
RL with episodic rewards [36]	Each node uses on policy Monte Carlo to update Q-values and policies at the end of each episode in order to select an honest next hop node for data transmission.	This model solves networking problems (e.g., routing) that are episodic in nature. Each episode consists of a set of consecutive actions needed to achieve an optimal action.	√					√				

4.1 RL model with suitability value

Vučević et al. [38] propose a RL model with suitability value for TRM S(1) applied to channel sensing P(1) in order to identify honest nodes for collaboration in CRNs. In this model, each node selects its neighbor nodes to collaborate by evaluating their suitability values, which are derived from the outcome of the final decisions. The suitability value is calculated using the softmax approach (see Section 2), and subsequently it is used to update the Q-function. Higher suitability value indicates higher probability that the node is chosen to collaborate. The purpose of the proposed RL model is to enable a node to monitor its neighbor nodes' behavior as time progresses in order to identify honest nodes in the presence of Byzantine A(1) and unintentional A(2) attacks. Hence, the RL model addresses the challenge of the dynamicity of malicious nodes' behavior C(1). The proposed RL model is a distributed model H(1.2), and it is embedded in each node. Each node makes decision independently without RL information exchange with neighbor nodes H(2.2). The proposed RL model has been shown to enhance the reliability of cooperation and specifically, it minimizes the probabilities of false alarm E(1) and missed detection E(3).

Table 6 shows the proposed RL model at node i for the TRM scheme in channel sensing. The state is not represented and so it is a stateless model. This means that the changes of the operating environment do not affect the node's action selection.

Table 6 RL model for channel sensing embedded in each node *i* [38].

Action	$a^i_{k\in K,\tau} \in A = \{0,1\}$, each action $a^i_{k\in K,\tau}$ represents a single collaborator node k for information exchange (i.e., sensing outcome), where $a^i_{k,\tau} = 1$ and $a^i_{k,\tau} = 0$ if collaborator node k is chosen and not chosen, respectively.
Reward	$r^i_{m\in M,\tau+1}(a^i_{k,\tau}) = N^i_{k,\tau}$, where $N^i_{k,\tau}$ represents the number of false alarms made by collaborator node k within a time window t. Node k experiences a false alarm whenever its sensing outcome indicates a channel with PUs' activities while the final decision shows otherwise.

Algorithm 2 presents the RL algorithm for the scheme at node i; while Figure 4 presents the flowchart of the algorithm. The flowchart is prepared for ease of comparison with the traditional RL algorithm presented in Section 2, as well as among the RL algorithms presented in this section. In this RL model with suitability value, the suitability value $0 \leq \phi_{k,\tau}^{i} \leq 1$ increases with Q-value, and hence, the updated Q-value $Q_{\tau+1}^{i}(a_{k,\tau}^{i})$ decreases with higher number of false alarms $r_{m,\tau+1}^{i}(a_{k,\tau}^{i})$ as shown in Equation (4) in Algorithm 2. The table also shows the computational and storage complexities of the RL algorithm for a single node i. Since the RL is executed in a distributed network with N SU nodes, the network-wide computational complexity is $O(N(N-1)|A|)$ and the network-wide storage complexity is $\leq (N(N-1)|A|)$. Please refer to Section 5.2 for the assumptions made and notations used in complexity analysis.

Algorithm 2 RL algorithm for channel sensing embedded in each node *i* [38] and its complexity.

RL algorithm		Complexity	
		Computational	Storage
Repeat			
1. Calculate suitability value $\phi_{k,\tau}^{i} = \frac{e^{Q_{k,\tau}^{i}(a_{k,\tau}^{i})}}{\sum_{j=1}^{K} e^{Q_{j,\tau}^{i}(a_{j,\tau}^{i})}}$		$O(1)$	
2. Choose action set $a_{\tau}^{i} = (a_{1,\tau}^{i}, a_{2,\tau}^{i}, \dots, a_{K,\tau}^{i})$			
3. Receive delayed reward $r_{m,\tau+1}^{i}(a_{k,\tau}^{i})$			
4. Update global reward $r_{m,\tau+1}^{i}(a_{\tau}^{i})$			
5. For $a_{k,\tau}^{i} \in a_{\tau}^{i}$, update Q-value for each collaborator node k: $Q_{\tau+1}^{i}(a_{k,\tau}^{i}) \leftarrow Q_{\tau}^{i}(a_{k,\tau}^{i}) + \alpha \cdot [r_{m,\tau+1}^{i}(a_{k,\tau}^{i}) - r_{m,\tau+1}^{i}(a_{\tau}^{i})] \cdot (1 - \phi_{k,\tau}^{i})$	(4)	$O(\|A\|)$	$\leq (\|A\|)$

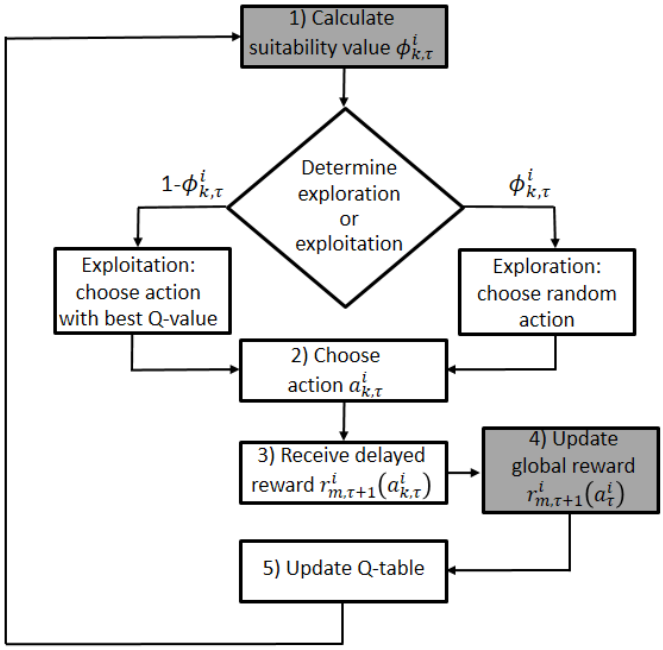

Figure 4: Flowchart of the RL algorithm with suitability value.

4.2 RL model with minimax Q-learning

Wang et al. [39] propose a RL model with minimax Q-learning for anti-jamming S(2) applied to channel access P(2) in order to maximize channel utilization in CRNs. This model learns the malicious nodes' actions or strategies in a zero-sum game [54] in which the malicious nodes' gains increase with decreasing gains from the honest nodes. This model updates the Q-value using a state value that is derived from the optimal policies by considering the worst scenario in which the malicious nodes adopt the best possible policies. The proposed RL model aims to enable SUs to maximize channel utilization in the presence of jamming attacks A(5). The malicious SUs may continually change their jamming strategies causing the honest SUs to also dynamically and strategically change their channel access policies in order to avoid the jammed channels. The malicious SU's objective is to optimize the effects of the attacks on SUs with limited jamming effort. Hence, the RL model addresses the challenge of the dynamicity of attack strategies C(3). The proposed RL model is a distributed model H(1.2), and it is embedded in each node. Each node makes decision independently without RL information exchange with

neighbor nodes H(2.2). The proposed RL model has been shown to increase SU spectrum utilization gain E(4), which is defined as a function of throughput, packet loss and delay.

Table 9 shows the proposed RL model at node i for the anti-jamming scheme in channel access. As for the malicious node h (which is not shown in Table 9), the action $a^h_{k\in K,\tau}$ represents malicious node h jamming spectrum k using previously un-attacked channels ($a^h_{k,D_1,\tau}$) or jamming previously jammed channels ($a^h_{k,D_2,\tau}$). Algorithm 3 presents the RL algorithm for the scheme at node i; while Figure 5 presents the flowchart of the algorithm. The table also shows the computational and storage complexities of the RL algorithm for a single node i. Since the RL is executed in a distributed network with N SU nodes, the network-wide computational complexity is $O(N(N-1)(|A|)$ and the network-wide storage complexity is $\leq (N(N-1)|S||A|)$.

Table 9 RL model for channel access embedded in each node *i* [39].

State	$s^i_{j\in J,\tau} = (P^i_{j,\tau}, g^i_{j,\tau}, H^i_{j,C,\tau}, H^i_{j,D,\tau}) \in S$, each state $s^i_{j\in J,\tau}$ consists of four sub-states: • $P^i_{j,\tau} = \{0,1\}$ represents the presence of PU in spectrum j in time slot $\tau = \Delta t$ ○ $P^i_{j,\tau} = 1$ ($P^i_{j,\tau} = 0$) indicates that PU is present (absent) • $g^i_{j,\tau} \in \{g_1, g_2, \cdots, g_J\}$ represents node i's gain in time slot $\tau = \Delta t$ ○ g_1(g_J) indicates the minimum (maximum) gain level • $H^i_{j,C,\tau}$ represents the number of jammed control channels in spectrum j in time slot $\tau = \Delta t$ • $H^i_{j,D,\tau}$ represents the number of jammed data channels in spectrum j in time slot $\tau = \Delta t$
Action	$a^i_{k\in K,\tau} = (a^i_{k,C_1,\tau}\, a^i_{k,D_1,\tau}, a^i_{k,C_2,\tau}\, a^i_{k,D_2,\tau}) \in A$, each action $a^i_{k\in K,\tau}$ consists of four sub-actions: • $a^i_{k,C_1,\tau}$ ($a^i_{k,D_1,\tau}$) represents node i transmits messages in control (data) channel selected from previously unjammed channel in spectrum j • $a^i_{k,C_2,\tau}$ ($a^i_{k,D_2,\tau}$) represents node i transmits messages in control (data) channel selected from previously jammed channel in spectrum j
Reward	$r^i_{m,\tau+1}(s^i_{j,\tau}, a^i_{k,\tau}, a^h_{k,\tau})$ represents the spectrum gain when the selected channel for transmission is unjammed in spectrum j

Algorithm 3 RL algorithm for channel access embedded in each node i [39] and its complexity.

RL algorithm	Complexity	
	Computational	Storage
Repeat		
1. Observe state $s_{j,\tau}^{i} \in S$		
2. Choose action $a_{k,\tau}^{i} \in A$		
3. Receive delayed reward $r_{m,\tau+1}^{i}(s_{j,\tau}^{i}, a_{k,\tau}^{i}, a_{k,\tau}^{h})$		
4. For $a_{k,\tau}^{i}$, update Q-value, optimal strategy, state value and learning rate for node i:		
$Q_{\tau+1}^{i}(s_{j,\tau}^{i}, a_{k,\tau}^{i}, a_{k,\tau}^{h}) \leftarrow (1-\alpha_{\tau}) Q_{\tau}^{i}(s_{j,\tau}^{i}, a_{k,\tau}^{i}, a_{k,\tau}^{h}) + \alpha_{\tau} \cdot [r_{m,\tau+1}^{i}(s_{j,\tau}^{i}, a_{k,\tau}^{i}, a_{k,\tau}^{h}) + \gamma \cdot V^{\pi^{i}}(s_{j,\tau+1}^{i})]$	$O(\lvert A\rvert)$	$\leq (\lvert S\rvert\lvert A\rvert)$
$\pi^{i,*}(s_{j,\tau}^{i}) \leftarrow \underset{\pi^{i}(s_{j,\tau}^{i})}{\operatorname{argmax}} \min_{\pi^{h}(s_{j,\tau}^{i})} \sum_{a_{k,\tau}^{i} \in A} Q(s_{j,\tau}^{i}, a_{k,\tau}^{i}, a_{k,\tau}^{h}) \cdot \pi^{i}(s_{j,\tau}^{i}, a_{k,\tau}^{i})$	$O(\lvert A\rvert)$	
$V^{\pi^{i}}(s_{j,\tau}^{i}) \leftarrow \min_{\pi^{h}(s_{j,\tau}^{i})} \sum_{a_{k,\tau}^{i} \in A} Q(s_{j,\tau}^{i}, a_{k,\tau}^{i}, a_{k,\tau}^{h}) \cdot \pi^{i,*}(s_{j,\tau}^{i})$	$O(\lvert A\rvert)$	
$\alpha_{\tau+1} \leftarrow \alpha_{\tau} \cdot \mu$	$O(\lvert 1\rvert)$	

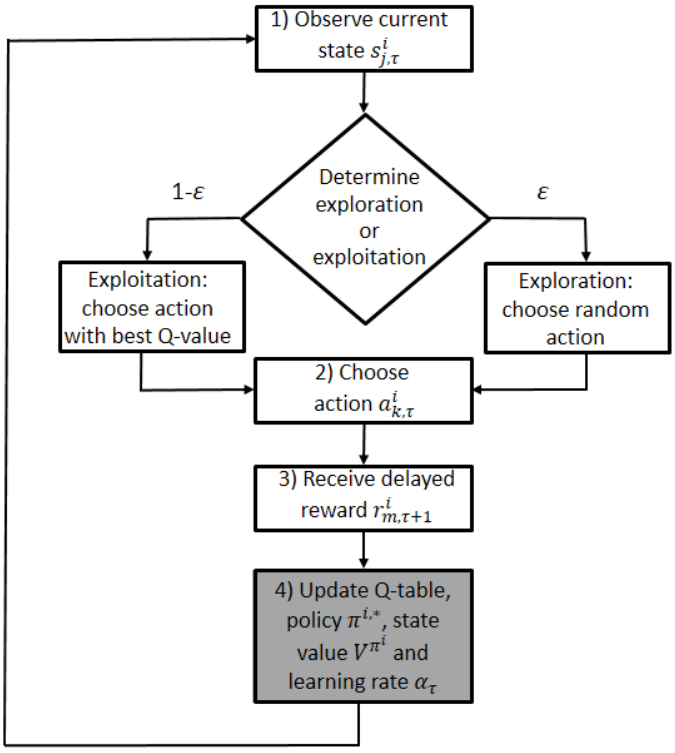

Figure 5: Flowchart of the RL algorithm with minimax Q-learning.

4.3 RL model with decreasing learning rate

Wu et al. [48] propose a RL model with decreasing learning rate for anti-jamming S(2) applied to channel access P(2) in order to maximize channel utilization in CRNs. This model derives the learning rate, which is the reciprocal of the number of updates for the Q-values of state-action pairs. Hence, the learning rate in this model reduces as time progresses. The purpose of the proposed RL model is to enable the SUs to maximize their long-term rewards in the presence of jamming attacks A(5) that may be launched at random, or with certain collaborative strategy, to attempt to jam the channels. The malicious SUs collaboratively launch attacks in order to minimize the honest SU's utilization gain. This may cause the honest SUs to dynamically and strategically change their operating channels in order to avoid the jammed channels. The malicious SU's objective is to maximize the adverse effects of the attacks on SUs. Hence, the RL model addresses the challenges of the dynamicity of attack strategies C(3) and the dynamicity of channel access C(4). The proposed RL model is a distributed model H(1.2), and it is embedded in each node. Each node makes decision independently without RL information exchange with its neighbor nodes H(2.2). The proposed RL model has been shown to increase spectrum utilization gain E(4).

Table 8 shows the proposed RL model at node i for the anti-jamming scheme in channel access. In contrast to the traditional reward representation, a different reward function is applied to calculate the delayed reward according to the different kinds of states observed and actions taken.

Table 8 RL model for channel access embedded in each node i [48].

State	$s^i_{j\in J,\tau} \in S = \{P, \ni, 1\}$, where • P indicates a PU occupies channel j • $\ni$ indicates a malicious SU jams channel j • 1 indicates a successful transmission in channel j
Action	$a^i_{k\in K,\tau} \in A = \{b, y\}$, where • b indicates node i chooses to switch to another channel • y indicates node i chooses to stay in its current operating channel
Reward	$r^i_{m,\tau+1}(s^i_{j,\tau}, a^i_{k,\tau})$ represents the channel gain for selecting an unjammed channel for transmission. Depending on the current state and action, there are four possible rewards $r^i_{m,\tau+1}$ as follows: • R when $a^i_{k,\tau} = y$ and $s^i_{j\in J,\tau} \neq P$ or $\ni$ • $R - C$ when $a^i_{k,\tau} = b$ and $s^i_{j\in J,\tau} \neq P$ or $\ni$ • $-L - C$ when $s^i_{j\in J,\tau} = \ni$ • $-C$ when $s^i_{j\in J,\tau} = P$ where R represents successful transmission gain, C represents the cost of switching to another channel, and L represents the cost incurred in choosing a jammed channel.

Algorithm 4 presents the RL algorithm for the scheme at node i; while Figure 6 presents the flowchart of the algorithm. The table also shows the computational and storage complexities of the RL algorithm for a single node i. Since the RL is executed in a distributed network with N SU nodes, the network-wide computational complexity is $O(N(N-1)|A|)$ and the network-wide storage complexity is $\leq (N(N-1)|S||A|)$.

Algorithm 4 RL algorithm for channel access embedded in each node i [48] and its complexity.

RL algorithm	Complexity	
	Computational	Storage
Repeat		
1. Observe state $s_{j,\tau}^{i} \in S$		
2. Choose action $a_{k,\tau}^{i} \in A$		
3. Receive delayed reward $r_{m,\tau+1}^{i}(s_{j,\tau}^{i}, a_{k,\tau}^{i})$		
4. Update Q-value for node i:		
$\alpha_\tau = \frac{1}{1 + \text{number of updates for } Q_\tau^i(s_{j,\tau}^i, a_{k,\tau}^i)}$	$O(\lvert 1 \rvert)$	
$Q_{\tau+1}^{i}(s_{j,\tau}^{i}, a_{k,\tau}^{i})$ $= \begin{cases} (1-\alpha)Q_\tau^i(s_{j,\tau}^i, a_{k,\tau}^i) & \text{when } (a_{k,\tau}^i = y, s_{j,\tau}^i \neq \{P, \Im\}) \\ +\alpha_\tau[r_{m,\tau+1}^i(s_{j,\tau}^i, a_{k,\tau}^i) + \gamma \cdot V^{\pi^i}(s_{j,\tau+1})], & \text{or } (a_{k,\tau}^i = b, s_{j,\tau}^i = \{P, \Im\}) \\ Q_\tau^i(s_{j,\tau}^i, a_{k,\tau}^i), & \text{when } (a_{k,\tau}^i = \{b, 1\}, s_{j,\tau}^i = \{P, \Im\}) \\ & \text{or } (a_{k,\tau}^i = b, s_{j,\tau}^i = \{P, \Im\}) \end{cases}$	$O(\lvert A \rvert)$	$\leq (\lvert S \rvert \lvert A \rvert)$
5. Update state value and policy for node i:		
$V^{\pi^i}(s_{j,\tau}^i)$ $= \begin{cases} \max[Q_{\tau+1}^i(s_{j,\tau}^i, b), Q_{\tau+1}^i(s_{j,\tau}^i, y)], & \text{when } s_{j,\tau}^i = 1 \\ Q_{\tau+1}^i(s_{j,\tau}^i, b), & \text{when } s_{j,\tau}^i = \{P, \Im\} \end{cases}$	$O(\lvert 1 \rvert)$	
$\pi^{i,*}(s_{j,\tau}^i) = \begin{cases} \underset{a \in \{b,y\}}{\text{argmax}} \; Q_{\tau+1}^i(s_{j,\tau}^i, a), & \text{when } s_{j,\tau}^i = 1 \\ b, & \text{when } s_{j,\tau}^i = \{P, \Im\} \end{cases}$	$O(\lvert 1 \rvert)$	

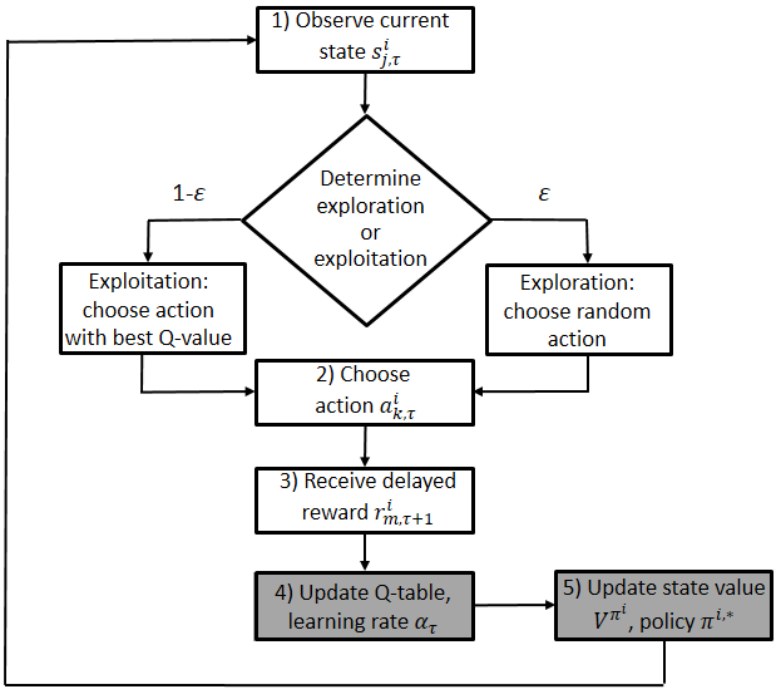

Figure 6: Flowchart of the RL algorithm with decreasing rate.

4.4 RL model with policy hill-climbing (PHC) and win-or-learn-fast (WoLF)

Lo et al. [49] propose a RL model with policy hill-climbing (PHC) and win-or-learn-fast (WoLF) for anti-jamming S(2) applied to channel access P(2) in order to maximize the CCC utilization in CRNs. This RL model aims to approximate the gradient ascent approach, which is a variant of the gradient descent approach [49], to adjust the step size of policy updates according to PUs' activities and malicious SUs strategies. Hence, when the PU's activity is low, WoLF increases the policy step size to ensure faster learning in order to avoid being attacked by the malicious SUs; and when the PU's activity is high (indicating low CCC availability), WoLF reduces the policy step size to delay malicious SUs' strategies [55] in order to ensure convergence to a greedy strategy. Next, the step size is used by PHC to update the policy using the step size given by WOLF. This RL model is multi-agent in nature because the honest SUs collaborate amongst themselves through message exchange (i.e., the common control information) using PHC and WoLF to achieve the maximum reward. The purpose of the proposed RL model is to enable the SUs to find an optimal control channel allocation strategy in the presence of jamming attacks

A(5). Hence, the RL model addresses the challenge of the data and control channel allocation C(5). The proposed RL model is a distributed model H(1.2), and it is embedded in each node. Each node makes decision independently without RL information exchange with neighbor nodes H(2.2). The proposed RL model has been shown to increase CCC utilization gain E(4). Table 9 shows the proposed RL model at node i for the anti-jamming scheme in channel sensing.

Table 9 RL model for channel sensing embedded in each node *i* [49].

State	$s^i_{j\in J,\tau} \in S = \{0,1\}$, where state $s^i_{j\in J,\tau} = 0$ and $s^i_{j\in J,\tau} = 1$ when channel j is not occupied and occupied by a PU, respectively.
Action	$a^i_{k\in K,\tau} \in A = \{0,\cdots,K\}$, where k represents the number of CCCs selected for packet transmission at time τ.
Reward	$r^i_{m,\tau+1}(s^i_{j,\tau}, a^i_{k,\tau}) = 1/U^i_{j=0,k,\tau}$, where $U^i_{j=0,k,\tau} \geq 0$ represents the number of valid CCCs which are unoccupied by PUs, jammed-free and common to SUs selected for packet transmission; $r^i_{m,\tau+1} = 0$ when there is no valid CCCs.

Algorithm 5 presents the RL algorithm for the scheme at node i; while Figure 7 presents the flowchart of the algorithm. Note that, δ is the step size for policy update where it brings a step closer to the optimal policy when the action taken maximizes the Q-value. The table also shows the computational and storage complexities of the RL algorithm. The table also shows the computational and storage complexities of the RL algorithm for a single node i. Since the RL is executed in a distributed network with N SU nodes, the network-wide computational complexity is $O(N(N-1)|A|)$ and the network-wide storage complexity is $\leq N(N-1)|S||A|)$. Note that, the complexity of PHC and WoLF are dependent on the algorithms themselves and hence, they are not considered in the complexity of this RL algorithm.

Algorithm 5 RL algorithm for channel sensing embedded in each node *i* [49] and its complexity.

RL algorithm	Complexity	
	Computational	Storage
Repeat 1. Observe state $s_{j,\tau}^{i} \in S = \{0,1\}$ 2. Choose action $a_{k\in K,\tau}^{i} \in A = \{1, \cdots, K\}$ 3. Receive delayed reward $r_{m,\tau+1}^{i}(s_{j,\tau}^{i}, a_{k,\tau}^{i})$ 4. For action $a_{k,t}^{i}$, update Q-value for node i: $Q_{\tau+1}^{i}(s_{j,\tau}^{i}, a_{k,\tau}^{i})$ $\leftarrow (1-\alpha) Q_{\tau}^{i}(s_{j,\tau}^{i}, a_{k,\tau}^{i})$ $+ \alpha[\, r_{m,\tau+1}^{i}(s_{j,\tau}^{i}, a_{k,\tau}^{i})$ $+ \gamma \max_{a\in A} Q_{\tau}^{i}(s_{j,\tau+1}^{i}, a_{k,\tau}^{i})]$ 5. If $r_{m,\tau+1}^{i}(s_{j,\tau}^{i}, a_{k,\tau}^{i}) \geq r_{m,TH}^{i}(a_{k,\tau}^{i})$, then $\delta = \delta_{\max}$ else apply WoLF to obtain δ where $r_{m,TH}^{i}$ is the reward threshold 6. Update policy for node i using PHC	$O(\|A\|)$	$\leq (\|S\|\|A\|)$

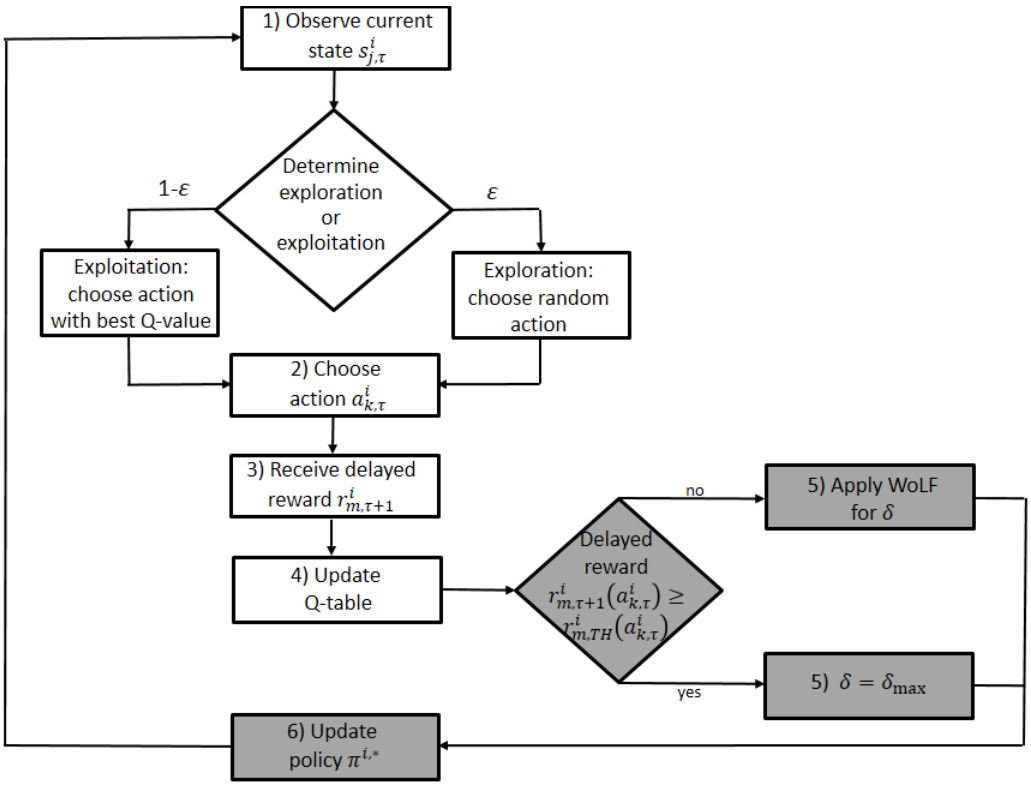

Figure 7: Flowchart of the RL algorithm with policy hill-climbing (PHC) and win-or-learn-fast (WoLF).

4.5 RL-based security enhancements in other wireless networks

4.5.1 Myopic approach: RL model with discount factor $\gamma = 0$

Mistry et al. [43] propose a RL model with discount factor $\gamma = 0$ for TRM S(1) applied to data reporting P(4) in order to manage the reputation values of decision fusion centers in WSNs (see Section 3.1.1.2). This model makes use of the most current reputation values of the upstream nodes for collaboration. The reputation value is derived from the number of accurate and inaccurate final decisions made by the upstream node within a time window $\tau = t$. Since the discount factor $\gamma = 0$, the Q-value is updated using delayed reward only. The purpose of the proposed RL model is to enable a node to monitor its upstream nodes' behavior as time progresses in order to identify honest upstream nodes, which play the role as decision fusion centers to aggregate sensing outcomes from downstream nodes, in the presence of random attacks A(3) and bias attacks A(4). Note that, with discount factor $\gamma = 0$, the future or discounted rewards are not considered and so only next-hop upstream nodes are considered in this model. RL has been

applied by downstream nodes to detect malicious next-hop upstream nodes. Hence, the RL model addresses the challenges of the dynamicity of malicious nodes' behavior C(1) and the dynamicity of network topology C(2). The proposed RL model is a centralized model H(1.1), and it is embedded in each potential downstream node. Each node makes decision independently without RL information exchange with neighbor nodes H(2.2). The proposed RL model has been shown to be efficient and accurate in updating and calculating the upstream nodes' Q-value, and hence, it increases the detection rate of malicious nodes E(2).

Table 10 shows the proposed RL model at node i for the TRM scheme in data reporting. The state is not represented and so it is a stateless model. This means that the changes of the operating environment do not affect the SU's action selection. Algorithm 6 presents the RL algorithm for the scheme at node i; while Figure 2, which is the generic flowchart, presents the flowchart of the algorithm. The table also shows the computational and storage complexities of the RL algorithm for a single node i. The RL is executed in a centralized network. The network-wide computational complexity is $O(N|A|)$ and the network-wide storage complexity is $\leq (N|A|)$.

Table 10 RL model for data reporting embedded in each downstream node *i* [43].

Action	$a^i_{k\in K,\tau} \in A_S = \{a^i_{k\in K,\tau} \in A \mid Q^i_{k,\tau}(a^i_{k,\tau}) > Q_{th}\}$, each action $a^i_{k\in K,\tau}$ represents a single upstream node k, where action $a^i_{k,\tau} = 1$ and $a^i_{k,\tau} = 0$ if upstream node k is chosen and not chosen, respectively. A_S represents a set of honest upstream nodes in which the reputation value of the upstream node $a^i_{k,\tau}$ is greater than a threshold Q_{th}.
Reward	$r^i_{m,\tau+1}(a^i_{k,\tau}) = (p^i_{k,\tau} - q^i_{k,\tau})/\,(p^i_{k,\tau} + q^i_{k,\tau} + 2)$, where $p^i_{k,\tau}$ and $q^i_{k,\tau}$ are the number of accurate and inaccurate final decisions, respectively. The reward indicates the reputation value of an upstream node k.

Algorithm 6 RL algorithm for data reporting embedded in each downstream node *i* [43] and its complexity.

RL algorithm	Complexity	
	Computational	Storage
Repeat 1. Choose action $a_{k,\tau}^{i}$ 2. Receive delayed reward $r_{m,\tau+1}^{i}(a_{k,\tau}^{i})$ 3. For action $a_{k,\tau}^{i}$, update Q-value for an upstream node k: $Q_{\tau+1}^{i}(a_{k,\tau}^{i}) \leftarrow (1-\alpha)Q_{\tau}^{i}(a_{k,\tau}^{i}) + \alpha \cdot r_{m,\tau+1}^{i}(a_{k,\tau}^{i})$	$O(\lvert A\rvert)$	$\leq (\lvert A\rvert)$

Similar myopic approach has been applied to [53] for TRM S(1) in data reporting P(4) in order to detect malicious nodes in WSNs (see Section 3.1.1.2). The purpose of the proposed RL model is to calculate the nodes' reputation values in order to identify honest nodes for collaboration in the presence of random attacks A(3). Hence, the RL model addresses the challenges of the dynamicity of malicious nodes' behavior C(1) and the dynamicity of network topology C(2). The proposed RL model is a centralized model H(1.1), and it is embedded in a decision fusion node. Each node makes decision independently without RL information exchange with neighbor nodes H(2.2). The proposed RL model has been shown to enhance the reliability of cooperation, specifically, to minimize probabilities of false alarm E(1) and missed detection E(3), as well as to maximize the detection rate of malicious nodes E(2).

Table 11 shows the proposed RL model at decision fusion node i for the TRM scheme in data reporting. The state is not represented and so it is a stateless model. This means that the changes of the operating environment do not affect the agent's action selection.

Table 11 RL model for data reporting embedded in node *i* [53].

Action	$a^i_{k\in K,t} \in A_S = \{a^i_{k\in K,t} \in A \mid Q^i_{k,t}(a^i_{k,t}) > Q_{th}\}$, each action $a^i_{k\in K,t}$ represents a single node k, where $a^i_{k,t} = 1$ and $a^i_{k,t} = 0$ if node k is not chosen and chosen, respectively.
Reward	$r^i_{k,t+1}(a^i_{k,t}) = e^{-(\varepsilon^i_{k,t}\mu_i/2\sigma^2\sigma_i^2)}$, where $\varepsilon^i_{k,t}$ is the relative error; and μ_i and σ_i are the mean and standard deviation of the predicted errors, respectively. The reward indicates the reputation value of neighbor node k.

4.5.2 RL model with episodic rewards

Maneenil et al. [36] propose a RL model with episodic rewards for TRM S(1) applied to routing P(3) in order to identify honest nodes for collaboration in WMNs (see Section 3.1.1.1). This model updates the Q-value with delayed reward only after an episode is completed. The purpose of the proposed RL model is to monitor next-hop neighboring node's behavior as time goes by in order to identify honest nodes in the presence of Byzantine attacks A(1). The identified honest nodes help to forward packets towards the destination node; while malicious nodes may discard the packets. The proposed RL model addresses the challenge of the dynamicity of the malicious nodes' behavior C(1). The proposed RL model is a distributed model H(1.2), and it is embedded in each node. Each node makes decision independently H(2.2). The proposed RL model has been shown to increase spectrum utilization gain E(4).

Table 12 shows the proposed RL model at node i for the TRM scheme in routing. Note that, an episode e_t is required to establish a route. This means a selected node for a route will only receive its reward after a route has been established. Therefore, an honest node has higher Q-value because it has been regularly chosen to forward packets. Algorithm 7 presents the RL algorithm for the scheme at a single node i; while Figure 8 presents the flowchart of the algorithm. Since the RL is executed in a distributed network with N SU nodes, the network-wide computational complexity is $O(N(N-1)(|A|)$ and the network-wide storage complexity is $\leq (N(N-1)|S||A|)$.

Table 12 RL model for routing embedded in each node *i* [36].

State	$s^i_{j\in J,\tau} \in S$ represents the reputation values of neighbor node $j \in J$, where J indicates all neighbor nodes of node i.
Action	The action $a^i_{k\in K,\tau} \in A$ represents the selection of a neighbor node k to forward packets towards destination, where $a^i_{k,\tau} = 1$ and $a^i_{k,\tau} = 0$ if node k is chosen and not chosen, respectively.
Reward	$r^i_{m,\tau+1}\left(a^i_{k,\tau}\right) = +1$ represents a constant value to be rewarded to all nodes within a route after an episode e_τ upon successful transmission.

Algorithm 7 RL algorithm for routing embedded in each node *i* [36] and its complexity.

RL algorithm	Complexity	
	Computational	Storage
Repeat		
1. Observe state $s^i_{j,\tau} \in S$		
2. Choose action $a^i_{k,\tau} \in A$		
3. Receive delayed reward $r^i_{m,\tau+1}\left(s^i_{j,\tau}, a^i_{k,\tau}\right)$		
4. Update Q-value for neighbor node k after an episode: $Q^i_{\tau+1}\left(s^i_{j,\tau}, a^i_{k,\tau}\right) \leftarrow \text{average}\left(r^i_{m,\tau+1}\left(s^i_{j,\tau}, a^i_{k,\tau}\right)\right)$	$O(\lvert A\rvert)$	$\leq (\lvert S\rvert\lvert A\rvert)$
5. Update policy for node i: $\pi^{i,*}\left(s^i_{j,\tau}, a^i_{k,\tau}\right) = \begin{cases} 1-\varepsilon+\frac{\varepsilon}{\lvert A\rvert}, & \text{when } a^i_{k,\tau} = \underset{a\in A}{\text{argmax}}\ Q^i_{\tau+1}\left(s^i_{j,\tau}, a\right) \\ \frac{\varepsilon}{\lvert A\rvert}, & \text{when } a^i_{k,\tau} \neq \underset{a\in A}{\text{argmax}}\ Q^i_{\tau+1}\left(s^i_{j,\tau}, a\right) \end{cases}$ where ε is the probability of selecting a random action	$O(1)$	

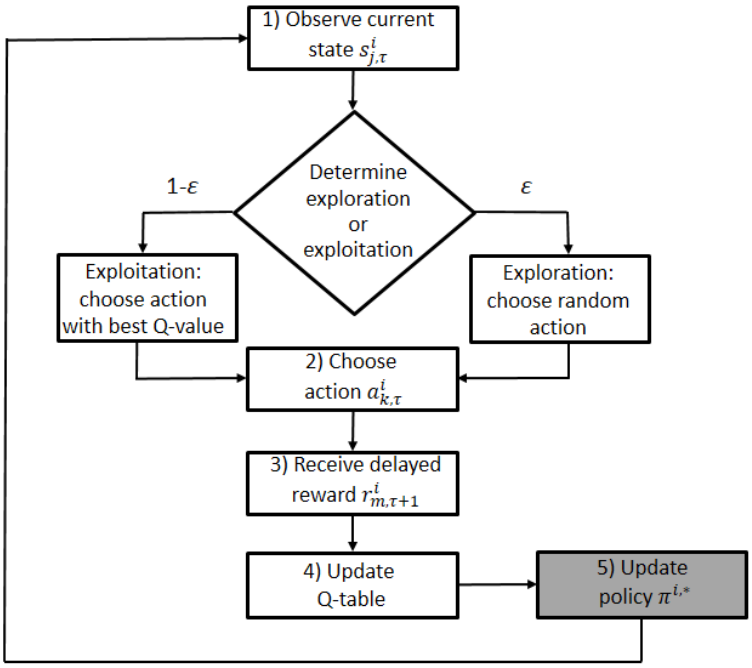

Figure 8: Flowchart of the RL algorithm with episodic reward.

5 RL Performance and Complexity Analysis

The effectiveness and usability of RL can be measured in terms of its performance enhancements and its complexity. Two scenarios, namely centralized and distributed CRNs, are presented for analysis. Section 5.1 tabulates the performance of various RL models in terms of different performance metrics, such as false positive, detection rate, missed detection and utilization gain. Section 5.2 discusses RL complexity in terms of computational and storage overhead complexities. Table 15 provides a summary of the RL complexities. The breakdown of the analysis can be found in sub sections 4.1 – 4.5 where the RL models and algorithms are presented in details.

5.1 Performance Enhancements

Table 13 provides a summary of the performance enhancements brought about by RL approaches in CRNs and other wireless networks. The performance metrics E(1) – E(4) have been previously discussed in detail in Section 3.6.

Table 13 Performance enhancements achieved by RL approaches.

Performance enhancement	RL Model					
	RL with suitability value [38]	RL with minimax Q-learning [39]	RL with decreasing learning rate [48]	RL with PHC WoLF [49]	RL with discount factor $\gamma = 0$ [43]	RL with episodic rewards [36]
E.1 Lower probability of false positive	√					
E.2 Higher detection rate					√	
E.3 Lower probability of missed detection	√					
E.4 Higher utilization gain		√	√	√		√

5.2 Complexity analysis

RL approaches incur computational and storage costs in terms of the time taken to calculate Q-values for the SUs, and the memory requirement needed to store Q-values. This section aims to discuss the complexity analysis of a general RL approach, which has been applied in security context.

5.2.1 Assumptions

In RL algorithms, SUs calculate the Q-values per time step. The values are subsequently used to detect malicious SUs. The following two types of general network models are considered:

- In a centralized network, the upstream node serves as the decision fusion center. It calculates the Q-value of each of the downstream nodes based on their actions taken. There are N nodes randomly distributed in the network.
- In a distributed network, all SUs observe their neighbors' action, as well as calculate and update their neighbors' Q-values. There are N SUs randomly distributed in the network with each SU having at most $N - 1$ SU neighbors.

For simplicity, henceforth, the nodes and SUs are referred as SUs.

5.2.2 Overview of a general RL model

In this article, we analyze RL algorithms with respect to computational and storage complexities associated with observing the state, taking an appropriate action and receiving a delayed reward in each learning cycle. The complexity analysis conducted in this section is inspired by similar investigation performed in [56].

5.2.3 Complexity Analysis

This section aims to investigate a general RL model with respect to computational and storage overhead complexities for the entire centralized and distributed networks, respectively. Table 14 describes the parameters used in the complexity analysis while Table 15 provides a summary of computational and storage overhead complexities for centralized and distributed networks. The breakdown of these complexities can be found in Algorithms 3–7.

We define the following terms:

- *Computational complexity* is the maximum number of times the RL algorithm is being executed in order to calculate the Q-values for all SUs in a network. The following calculation of complexities is for a simple and generic RL algorithm. In a centralized network, upon receiving actions from N SUs, the SU (fusion center) calculates and updates their respective Q-values, so the computational complexity is $O(N|A|)$. In a distributed network, each SU receives at most $N-1$ actions, so the computational overhead is $O((N-1)|A|)$ at each SU; hence, with N SUs in the network, the computational complexity is $O(N(N-1)|A|)$.
- *Storage overhead complexity* is the amount of memory needed for the storage of the values during the course of a RL algorithm execution. Suppose, each SU maintains a table that keeps track of the Q-values. In a centralized network, each SU with non-stateless RL model, has $|S||A|$ Q-values, so the storage overhead complexity is $\leq N(|S||A|)$. In a distributed network, each SU with non-stateless RL model, has $|S||A|$

Q-values, so the storage overhead is $\leq (N-1)(|S||A|)$ at each SU; hence, with N SUs in the network, the storage overhead complexity is $\leq N(N-1)(|S||A|)$.

Table 14 Parameters for complexity analysis.

Parameter	Description
$\|S\|$	Number of states
$\|A\|$	Number of actions for each state
N	Number of SUs in the network.

Table 15 Network-wide complexities of RL for centralized and distributed CRNs.

Network Model	Reinforcement learning model	Complexities	
		Computational	Storage
Centralized	RL with discount factor $\gamma = 0$ [43]	$O(N\|A\|)$	$\leq (N\|A\|)$
	[53]		$\leq (N\|S\|\|A\|)$
Distributed	RL with suitability value [38]	$O(N(N-1)\|A\|)$	$\leq (N(N-1)\|A\|)$
	RL with minimax Q-learning [39]		$\leq (N(N-1)\|S\|\|A\|)$
	RL with decreasing learning rate [48]		
	RL with PHC and WoLF [49]		
	RL with episodic rewards [36]		

6 Guidelines and Design Considerations for the Application of RL to Security Enhancement in CRNs

This section presents guidelines and design considerations for application of RL to security enhancement in CRNs.

6.1 Guidelines for the application of RL to CRNs

When considering application of RL for security enhancement in CRNs, a problem or open issue at hand needs to be identified and well understood. This includes the objectives and purposes, as well as the problem statement and research questions applicable to the problem. Subsequently, the following questions need to be answered. We present guidelines for the application of RL to CRNs in the light of a sample case study [39] that we will refer to throughout this subsection. In [39], RL with minimax Q-learning for anti-jamming is applied to channel access in order to maximize channel utilization in CRNs. Next, we define the state, action and reward for the anti-jamming scheme as follows:

a) *Defining state.* What are the decision making factors that an agent observe from the operating environment? For instance, in [39], the objectives are to counter jamming attacks in order to maximize spectrum utilization. Therefore, the agent represents the states with the presence of PU in spectrum j, node i gain (i.e., throughput), the number of jammed control channels in spectrum j, and the number of jammed data channels in spectrum j. Upon observing the state, the agent makes decision on its action based on the state.

b) *Defining action.* What are the possible actions that an agent can take to maximize its rewards? For instance, in [39], with respect to the objective of avoiding jammers, the agent must choose the available control and data channels to transmit. It is expected that, by choosing an action in an intelligent manner, the agent chooses an unjammed channel. This allows the agent to receive higher rewards.

c) *Defining reward.* What is the expected delayed reward received (or performance enhancement enjoyed) by an agent after it has taken an action in the state? For instance, in [39], the delayed reward is the spectrum gain when the agent selects a channel that is unjammed.

d) *Choosing an algorithm.* What are the objectives of the algorithm? The main objective is to help SUs to learn the dynamic attack strategies from malicious SUs, who tend to optimize their attacks. Therefore, SUs must learn to take optimal actions in the presence of worst case

attacks from malicious SUs. Hence, minimax Q-learning algorithm is chosen. The rest of the RL algorithms shown in Table 5 can be selected based on the main objective of the security scheme.

6.2. Design considerations for the application of RL to CRNs

This section presents some considerations that can be taken into account when designing a RL model for security enhancement in CRNs.

6.2.1 Representation of state, action and reward types

In CRNs, where PUs' activities can be dynamic and unpredictable, effective application of Q-learning algorithm to channel sensing and channel access requires some security considerations to be met with regards to defining states and actions, and assigning reward values. In some operating environment where only one state exists, the agent independently selects and performs an action, and receives a reward. The agents then updates its policy based on this reward, and the next iteration starts. Such an environment is static, i.e., no state transition occurs as can be seen in [38, 43]. In [38, 43], the SUs do not consider the state of the operating environment such as PU existence. On the other hand, RL allows the operating environment to be expressed in a comprehensive yet condensed format to represent the real world environment as seen in [39] in order to learn the hostile environment. Additionally, in channel sensing, instead of a fixed reward value, the value may be assigned according to the current scenario or activity in order to avoid or detect malicious SUs. For instance, the rewards in RL models [38, 46] are derived from the number of channels accurately sensed and the number of valid CCCs selected, respectively. This reflects the currency of the given reward. Such reward variable can also be seen in RL models [39, 48].

6.2.2 Multi-agent RL

The RL models discussed in this article are single-agent RL (SARL) except for [49], which is a multi-agent RL (MARL) approach. Previous works [57 – 60] have shown that SARL performance in partially observable, non-Markovian and multi-agent systems can be unsatisfactory. For instance, policy-gradient methods [59, 60] have been shown to outperform RL, where the policy-gradient approach has been shown

to be more efficient in partially observable environments since it searches directly for optimal policies in the policy space. While some work has been done in MARL environment [61, 62], such as designing learning policies for CRNs, some aspect of security can be taken into consideration, such as incorporating TRM into the MARL environment to provide an additional layer of detection for malicious SUs.

6.2.3 Learning rate

In an operating environment where attackers' number and strategies are dynamic, learning rate in RL models can be adjusted according to the current scenario. An appropriate choice of learning rate will enable the SU to learn accurately from the operating environment. For instance, in [39], the learning rate decreases as the SU has learnt enough to exploit the operating environment, while in [46], the RL algorithm makes use of WoLF to adjust the learning rate based on the PU's activity.

6.2.4 Discount factor

In an operating environment where the future reward accumulated by an agent is considered as important, the discount factor may be used for the purpose [17]. Higher value of the discount factor indicates stronger emphasis on maximizing the long-term reward. The RL parameter can be adjusted accordingly depending on the operating scenario emphasis. For instance, in [43, 53], the short-term reward ($\gamma = 0$) is used to detect malicious nodes, while in [48], the long term reward ($\gamma = 0.95$) is used.

6.2.5 State space explosion

State space explosion occurs when the RL algorithm is applied to a large-scale operating environment (which has an increased number of states and the size of each states). Such increase can incur an exponential growth in the learning time due to slower convergence and increased computational problems in terms of memory and speed. This is mainly due to the size of the Q-tables which increases growth exponentially. Such growth can have an adverse effect on RL algorithm's performance, as it may not be able to detect malicious SU in an efficient manner. Hence, a RL algorithm of this capacity and capability may need to find an alternative approach. In [63], batch RL is used to solve the state space explosion problem. The RL batch uses algorithms such as Fitted Q-Iteration [64] and Least-Squares Policy Iteration

[65] to store state-action-reward tuples and process them in batches.

7 Open Issues

This section discusses some open issues associated with the application of RL to security enhancement scheme that can be pursued.

7.1 Balancing the trade-off between exploration and exploitation

Yen et al. [66] show that, in a dynamic operating environment, it is possible for a Q-learning agent to be bounded in a small area of state space due to exploitation, and this may result in its inability to detect attackers' behavior, such as honest nodes that turn malicious C(1). While it may be desirable to explore in a dynamic operating environment in order to increase the agent's flexibility to adapt to the changing environment, pure exploration may degrade the agent's learning capability [67]. Hence, the main challenge in exploration and exploitation is to find a balanced trade-off at the shortest possible time in order to achieve the maximum reward [68], such as higher detection rate E(2) and higher utilization gain E(4); and subsequently to incorporate the mechanism into RL algorithms. Traditional learning policies such as ε-greedy, Boltzmann exploration (softmax), simulated annealing and probability matching could also be studied and used for comparison. Note that, in such an investigation, it is also important to find the convergence results of the mechanism.

7.2 Determining the learning rate α value

While RL model has been shown to be effective in minimizing the probability of false positive [38] E(1), it has been noted in [43] that an increase in learning rate α value may increase false positives. For instance, if $\alpha = 1$, the agent considers only the most current Q-value, which may not be optimal or accurate especially when the agent is exploring the state space. The increase in false positives is detrimental to RL as it falsely reports attacks or malicious nodes when there are none, leading to

inaccurate decisions. On the other hand, a decrease in learning rate α value increases the missed detections on malicious nodes. For instance, if $\alpha = 0$, the agent relies on the previous or old Q-values only, which again may not be accurate especially in a dynamic operating environment, where nodes' behavior may vary as time progresses, such as honest nodes that turn malicious C(1). Hence, it is important to find an acceptable value for α so that false positive and missed detection rate can be at their lowest optimal in order to provide a more accurate and timely solution. Even-Dar et al. [69] show a relationship between the learning rate and the convergence rate, and their work can be further investigated to find the lower and upper bounds of these values in Q-learning algorithms.

7.3 Applying the right policy

RL algorithms are expected to detect malicious nodes in wireless networks in the shortest possible time (or with the highest possible convergence rate) in most sizes of state spaces. An efficient RL algorithm needs to adopt a suitable policy that reflects the current status of the operating environment, where an honest node may turn malicious, and vice-versa C(1). The update of a policy can be performed at every time instant (called immediate policy) or at the end of each epoch, which consists of a number of time instants (called epoch policy). Higher convergence rate indicates lower number of time instants and epochs needed to achieve the optimal policy. Using the epoch policy [70], the efficiency of RL algorithms may be influenced by the policy update frequency. However, using the immediate policy may also incur higher learning time and this may decrease the convergence rate. Hence, further work could be carried out to explore suitable policies for RL algorithms in various operating environment settings in order to improve convergence rate.

7.4 Ameliorating the curse of dimensionality

While the majority of current works consider small state space only, the state space in a real-world environment may be large and dynamic in nature C(2) and C(4). For instance, multi-agent reinforcement learning (MARL) [40] faces the curse of dimensionality problem, which results in the exponential growth of the state-action pairs, when the number of agents increases. As a result, the computational complexity

of the RL algorithm, which is the number of times the algorithm needs to be executed, increases exponentially too [40, 71]. This may cause the algorithm to perform poorly leading to a longer time required to detect malicious nodes in the network. Investigation could be carried out to incorporate feature selection method, which is a preprocessing step in RL to remove the unimportant features in the state space [55, 72]. This step reduces the dimension of data and it may improve the speed of detection of malicious nodes.

7.5 Applying the right epoch time

Given the intrinsic nature of RL where the delayed reward is received at the end of the next epoch time in the epoch policy, it is worth studying the duration of each time epoch in which the attackers may leverage a longer epoch time to dynamically change their behavior C(1) and attack strategies C(3). In the epoch policy, the update is only done at the end of each epoch to minimize the computational complexity [70]. However, longer duration of each epoch may inadvertently open up opportunities for the attackers to improve their attack strategies. Further work can be carried out to study the implication and various duration of each epoch, and the maximum allowable epoch time in order to reduce the number of attacks.

7.6 Investigating the reward value assignment based on the severity of attacks

An important component of a RL-based security enhancement scheme is the construction of the reward function. By appropriately defining the reward function, a RL scheme can help increase the detection rate of malicious nodes while reducing the probabilities of false alarm and missed detection. In [73], an investigation was conducted on when, what and how much to reward in RL. *When* determines the moment which may be the end of each epoch, subtask or other interval of task, *what* is the objective function such as duration and accuracy, and *how much* determines the magnitude of a reward. Similar study can be carried out to investigate the feasibility of assigning a reward value based on the consequence or impact of the attacks. In [74, 75], the authors constructed their reward functions based on the characteristic of the states. Further work can also be done to assign rewards based on the severity of

attacks. For instance, when an agent experienced an intelligent attack A(6), it should receive much lesser reward than that of unintentional attacks A(2).

7.7 Using RL to predict attacks

In Ezirim et al. [47], the attackers launch intelligent attacks A(6) by leveraging RL to maximize the potency of attack. For instance, in CRNs, attackers launch RL-based Sybil attacks (a form of denial-of-service attacks) to learn the optimal number of false sensing outcomes to be sent to a fusion center without being detected as malicious. Such attacks may affect the accuracy of the final decisions on sensing outcomes, which may result in higher rate of false positives. Further work could be carried out to counter such intelligent attacks by using RL to predict the imminent attacks in the operating environment based on the SUs' activities.

7.8 Applying cooperative agents in MARL

A promising approach to detect malicious nodes in CRNs is to get the neighboring nodes to collaborate. As shown in [76], cooperative agents in multi-agent-based RL approach can significantly improve the performance of a joint task, and such cooperation has been shown to speed up the learning process and subsequently converge sooner as compared to independent agents [77]. Further work could be carried out to measure the effectiveness of applying cooperative agents in CRNs. In addition to measuring the speed of the detection of malicious nodes in a dynamic operating environment, performance metrics, such as higher detection rate E(2) could also be analyzed to ensure that the mechanism offers optimum speed of detection.

8 Conclusions

In this article, we presented an extensive review on the use of reinforcement learning (RL) to achieve security enhancement in cognitive radio networks (CRNs), as well as other wireless networks. RL is an unsupervised and intelligent approach that enables an agent to observe and learn about the static or

dynamic operating environment in the absence of guidance, feedback or the expected response from supervisors, and subsequently make decisions on action selection in order to achieve optimal or near-optimal system performance. RL-based security enhancement schemes in CRNs are capable of learning new security attacks and to detect previously learned ones. RL-based security enhancement schemes have been successfully applied in a number of diverse problems, such as channel sensing, channel access, routing and data sensing/ reporting. This article presents the performance enhancements of security enhancement schemes achieved by RL: lower probability of false positive and missed detection, higher detection rate, and higher utilization gain. Various RL models, such as RL model with suitability value, RL model with minimax Q-learning, and RL model with policy hill-climbing and win-or-learn-fast have been studied. This article also presents a complexity analysis of these RL models, and discusses a number of open issues associated with RL, such as balancing trade-off between exploration and exploitation, determining the learning rate value, and ameliorating the curse of dimensionality.

Acknowledgment

This work was supported by the Ministry of Education Malaysia under Fundamental Research Grant Scheme (FRGS) FRGS/1/2014/ICT03/SYUC/02/2. Mee Hong Ling, Kok-Lim Alvin Yau and Qiang Ni were also funded under the Small Grant Scheme (Sunway-Lancaster), grant agreement number SGSSL-FST-CSNS-0114-05 and PVM1204.

References

[1] Mitola J, Maguire GQ. Cognitive radio: making software radios more personal, *IEEE Personal Communications* 1999; 6(4): 13–18.
[2] Akyildiz IF, Lee W, Vuran MC, Mohanty S. NeXt generation/dynamic spectrum access/cognitive radio wireless networks: a survey, *Journal of Computer Networks* 2006; 50(13): 2127–2159. DOI=10.1016/j.comnet.2006.05.001
[3] Lo BF, Akyildiz IF, Al-Dhelaan AM. Efficient Recovery Control Channel Design in Cognitive Radio Ad Hoc Networks, *IEEE Transactions on Vehicular Technology* 2010; 59(9):4513–4526.
[4] Fragkiadakis A, Tragos E, Askoxylakis I. A survey on security threats and detection techniques in cognitive radio networks, *IEEE Communications Surveys & Tutorials*, 2013; 15(1): 428– 445.
[5] Bhattacharjee S, Sengupta S, Chatterjee M. Vulnerabilities in Cognitive Radio Networks: A Survey, *Computer Communications* 2013; 36(13):1387-398.
[6] Sharma R, Rawat D. Advances on Security Threats and Countermeasures for Cognitive Radio Networks: A Survey, in: *IEEE Communications Surveys & Tutorials,* doi: 10.1109/COMST.2014.2380998.
[7] Sutton RS, Barto AG. Reinforcement Learning: An Introduction, The MIT Press, Cambridge, Massachusetts, 1998.
[8] Hamdaoui B, Venkatraman P, Guizani M. Opportunistic exploitation of bandwidth resources through reinforcement learning, in: *IEEE Global Telecommunications Conference (GLOBECOM '09)*, December 2009.

[9] Yau KLA, Komisarczuk P, Teal PD. Applications of reinforcement learning to cognitive radio networks, in: *IEEE International Conference on Communications Workshops (ICC)*, May 2010.
[10] Reddy Y. Detecting primary signals for efficient utilization of spectrum using Q-learning, in: *5th International Conference on Information Technology: New Generations (ITNG '08)*, April 2008.
[11] Li M, Xu Y, Hu J. A Q-learning based sensing task selection scheme for cognitive radio networks, in: *International Conference on Wireless Communications Signal Processing (WCSP '09)*, November 2009.
[12] Yao Y, Feng Z. Centralized channel and power allocation for cognitive radio networks: A Q-learning solution, in: *Future Network and Mobile Summit*, June 2010.
[13] Venkatraman P, Hamdaoui B, Guizani M. Opportunistic bandwidth sharing through reinforcement learning, *IEEE Transactions on Vehicular Technology* 2010; 59(6): 3148 –3153.
[14] Jiang T, Grace D, Mitchell P. Efficient exploration in reinforcement learning-based cognitive radio spectrum sharing, *IET Communications* 2011; 5(10):1309 –1317.
[15] Lunden J, V. Koivunen V, Kulkarni S, Poor H. Reinforcement learning based distributed multiagent sensing policy for cognitive radio networks, in: *IEEE Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN '11)*, May 2011.
[16] A. Galindo-Serrano and L. Giupponi. Distributed Q-learning for aggregated interference control in cognitive radio networks, *IEEE Transactions on Vehicular Technology* 2010; 59(4):1823 –1834.
[17] Parvin, S, Hussain FK, Hussain OK, Han S, Tian B, Chang E. Cognitive radio network security: A survey, *Journal of Network and Computer Applications* 2012; 35(6):1691–1708.
[18] Kaelbling LP, Littman ML, Moore AW. Reinforcement learning: A survey, *Journal of Artificial Intelligence Research* 1996; 4:237–285.
[19] He A, Bae KK, Newman TR, Gaeddert J, Kim K, Menon R, Morales-Tirado L, Neel JJ, Zhao Y, Reed JH, Tranter WH. A survey of artificial intelligence for cognitive radios, *IEEE Transactions on Vehicular Technology* 2010; 59(4):1578–1592.
[20] Bertsekas DP. Dynamic Programming and Optimal Control. Athena Scientific: 1995.
[21] Shokri M. Knowledge of opposite actions for reinforcement learning, *Journal of Applied Soft Computing* 2011; 11: 4097–4109.
[22] Leng J, Lim CP. Reinforcement learning of competitive and cooperative skills in soccer agents, *Journal of Applied Soft Computing* 2011; 11:1353–1362.
[23] Clancy TC, Goergen N. Security in Cognitive Radio Networks: Threats and Mitigation, in: *3rd International Conference on Cognitive Radio Oriented Wireless Networks and Communications, (CrownCom)*, May 2008.
[24] Dabcevic K, Betancourt A, Marcenaro L, Regazzoni, CS. Intelligent cognitive radio jamming-a game-theoretical approach, *EURASIP Journal on Advances in Signal Processing* 2014; (1)171.
[25] Pietro RD, Oligeri G. Jamming mitigation in cognitive radio networks, *IEEE Network* 2013; 27**(**3):10–15.
[26] Chen C, Song M, Xin CS, Backens, J. A game-theoretical anti-jamming scheme for cognitive radio networks, *IEEE Network* 2013; (27)3:22–27, DOI: 10.1109/MNET.2013.6523804.
[27] Yuan Z, Niyato D, Li H, Song JB, Han Z. Defeating Primary User Emulation Attacks Using Belief Propagation in Cognitive Radio Networks, *IEEE Journal on Selected Areas in Communications* 2012; (30)10:1850–1860, DOI: 10.1109/JSAC.2012.121102.
[28] Tan Y, Sengupta S, Subbalakshmi KP. Primary user emulation attack in dynamic spectrum access networks: a game-theoretic approach, *IET Communications* 2012; (6)8:964 –973, DOI: 10.1049/iet-com.2010.0573.
[29] Nguyen-Thanh N. Ciblat P, Pham AT, Nguyen V. Attack and Surveillance Strategies for Selfish Primary User Emulator in Cognitive Radio Network, in: *the 2nd IEEE Global Conference on Signal and Information Processing*, December 2014.
[30] Kapetanakis S, Daniel Kudenko D, Strens MJA. Reinforcement learning approaches to coordination in cooperative multiagent systems, In *Adaptive Agents and Multi-Agent Systems*. Springer Berlin/Heidelberg, January 2003.
[31] Yau K-LA, Komisarczuk P, Teal PD. Security aspects in the cognition cycle of distributed cognitive radio networks: a survey from a multi-agent perspective, *International Journal of Ad Hoc and Ubiquitous Computing* 2013; 12(3):157–176.
[32] Bkassiny M, Yang Li, Jayaweera, SK. A Survey on Machine-Learning Techniques in Cognitive Radios, *IEEE Communications Surveys & Tutorials* 2013; (15)3:1136–1159. DOI: 10.1109/SURV.2012.100412.00017
[33] Yau KLA, Poh GS, Chien SF, Al-Rawi HAA. Application of Reinforcement Learning in Cognitive Radio Networks: Models and Algorithms, *The Scientific World Journal* 2014; DOI:10.1155/2014/209810.
[34] Watkins CJCH. Learning from Delayed Rewards, *Cambridge University*, 1989.
[35] Ghasemi A, Sousa ES. Spectrum sensing in cognitive radio networks: requirements, challenges and design trade-offs, *IEEE Communications Magazine* 2008; 46(4):32–39.
[36] Maneenil K, Usaha W. Preventing malicious nodes in ad hoc networks using reinforcement learning, in: *2nd International Symposium on Wireless Communication Systems*, September 2005.
[37] Sodagari S, Clancy TC. An anti-jamming strategy for channel access in cognitive radio networks, in: *Proceedings of the 2nd International Conference on Decision and Game Theory for Security, Springer-Verlag: College Park, Maryland* 2011; 34–43.
[38] Vučević N, Akyildiz IF, Pérez-Romero J. Dynamic cooperator selection in cognitive radio networks, Ad Hoc Networks 2012; 10(5):789–802.
[39] Wang B, Wu Y, Liu KJR, Clancy TC. An Anti-Jamming Stochastic Game for Cognitive Radio Networks, *IEEE Journal on Selected Area in Communications* 2011; 29(4):877–889.
[40] Busoniu L, Babuska R, and Schutter BD. A Comprehensive Survey of Multiagent Reinforcement Learning, *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews* 2008; 38(2):156–172.
[41] Akyildiz IF, Su W, Sankarasubramaniam Y, Cayirci E. A survey on sensor networks, *IEEE Communications Magazine* 2002, 40(8):102–114.
[42] Fragkiadakis A, Vangelis A, Tragos E. Securing Cognitive Wireless Sensor Networks: A Survey, *International Journal of Distributed Sensor Networks*, 2014.
[43] Mistry O, Gursel A, Sen S. Comparing trust mechanisms for monitoring aggregator nodes in sensor networks, in: *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems* 2009; 985–992.
[44] Awerbuch B, Curtmola R, Holmer D, Nita-Rotaru C, Rubens H. Mitigating byzantine attacks in ad hoc wireless networks, *Department of Computer Science, Johns Hopkins University, Technical Report Version 1*, 2004.

[45] Ling MH, Yau KLA, Poh GS. Trust and reputation management in cognitive radio networks: a survey, *Security Communication Networks*, 2014; 7(11): 2160–2179.
[46] Lo BF. A survey of common control channel design in cognitive radio networks, *Physical Communication* 2011; 4(1):26–39.
[47] Ezirim K, Troja E, Sengupta S. Sustenance against RL-based Sybil attacks in Cognitive Radio Networks using Dynamic Reputation systems, in: *Proceedings of IEEE Military Communications* (*MILCOM), 2013.*
[48] Wu Y, Wang B, Liu KJR, Clancy TC. Anti-Jamming Games in Multi-Channel Cognitive Radio Networks, *IEEE Journal on Selected Areas in Communications* 2012; 30(1):4–15.
[49] Lo BF, Akyildiz IF. Multiagent Jamming-Resilient Control Channel Game for Cognitive Radio Ad Hoc Networks, in: *Proceedings of IEEE International Conference on Communications*, 2012.
[50] Zhang Z, Naït-Abdesselam F, Ho PH, Kadobayashi Y. Toward cost-sensitive self-optimizing anomaly detection and response in autonomic networks, *Computers & Security* 2011; 30(6/7):525–537.
[51] Navda V, Bohra A, Ganguly S, Rubenstein D. Using Channel Hopping to Increase 802.11 Resilience to Jamming Attacks, in: *26th IEEE International Conference on Computer Communications (INFOCOM)* 2007.
[52] Wood AD, Stankovic JA. Denial of service in sensor networks, *Computer* 2002; 35(10):54–62.
[53] Mukherjee P, Sen S. Comparing Reputation Schemes for Detecting Malicious Nodes in Sensor Networks, *Computer Journal* 2011; 54(3):482–489.
[54] Hwnag K, Chiou J, Chen T. Reinforcement Learning in Zero-Sum Markov Games for Robot Soccer Systems, in: *Proceedings of the 2004 IEEE International Conference on Networking, Sensing & Control* 2004.
[55] Bowling M, Veloso M. Multiagent learning using a variable learning rate, *Artificial Intelligence* 2002; 136:215–250.
[56] Geramifard A, Walsh TJ, Tellex S, Chowdhary G, Roy N, How JP. A tutorial on linear function approximators for dynamic programming and reinforcement learning 2013; DOI: 10.1561/2200000042.
[57] Claus C, Boutilier C. The dynamics of reinforcement learning in cooperative multiagent systems, in: *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, July 1998.
[58] Croon GD, Dartel MFV, Postma EO. Evolutionary learning outperforms reinforcement learning on non- markovian tasks, in: *Workshop on Memory and Learning Mechanisms in Autonomous Robots, 8th European Conference on Artificial Life*, 2005.
[59] Sutton R, Mcallester D, Singh S, Mansour Y. Policy gradient methods for reinforcement learning with function approximation, in: *Proceedings of the 12th conference on Advances in Neural Information Processing Systems (NIPS '99)*. 2001.
[60] Baxter J, Bartlett PL. Infinite-horizon policy-gradient estimation, *Journal of Artificial Intelligence Research* 2001; (15):319–350.
[61] Schaar M, Fu F. Spectrum access games and strategic learning in cognitive radio networks for delay-critical applications, in: *Proceedings of the IEEE* 2009; 97(4):720–740.
[62] Wang B, Liu KR, Clancy T. Evolutionary cooperative spectrum sensing game: how to collaborate? *IEEE Transactions on Communications* 2010; 58(3):890 –900.
[63] Lange S, Gabel T, Riedmiller M. Batch reinforcement learning 2010; (12):45–73 Springer Berlin Heidelberg.
[64] Ernst D, Geurts P, Wehenkel L. Tree-based batch mode reinforcement learning, *Journal of Machine Learning Research* 2005; pp. 503–556.
[65] Lagoudakis MG, Parr R. Least-squares policy iteration, *The Journal of Machine Learning Research* 2003; (4):1107–1149.
[66] Yen G, Yang F, Hickey T. Coordination of Exploration and Exploitation in a Dynamic Environment, *International Journal of Smart Engineering System Design* 2010; 4(3):177–182.
[67] Fard SMH, Hamzeh A, Hashemi S. Using reinforcement learning to find an optimal set of features, *Computers & Mathematics with Applications* 2013; 66(10):1892–904.
[68] Kulkarni P. Reinforcement and systemic machine learning for decision making, John Wiley & Sons, Inc., 2012.
[69] Even-Dar E, Mansour Y. Learning rates for Q-learning, *Journal of Machine Learning Research* 2004; 5:1–25.
[70] Zajdel R. Epoch–Incremental Reinforcement Learning Algorithms, *International Journal of Applied Mathematics and Computer Science* 2013; 23(3):623–635.
[71] Gutierrez-Osuna R. I*ntroduction to Pattern Analysis,* Texas A&M University, 2002.
[72] Gaudel R, Sebag M. Feature Selection as a One-Player Game, *International Conference on Machine Learning* 2010.
[73] Janssen CP, Gray WD. When, What, and How Much to Reward in Reinforcement Learning-Based Models of Cognition, *Cognitive Science* 2012; 36(2):333–358.
[74] Vanhulsel M, Janssens D, Wets G. Calibrating a new reinforcement learning mechanism for modeling dynamic activity-travel behavior and key events; 2007.
[75] Charypar D, Nagel K. Q-learning for flexible learning of daily activity plans, *Journal of the Transportation Research Board* 2005; 1935:163–169.
[76] Tan M. Multi-agent reinforcement learning: Independent vs. cooperative agents, in: *Proceedings of the 10th International Conference on Machine Learning* 1993.
[77] Berenji HR, Vengerov. Advantages of cooperation between reinforcement learning agents in difficult stochastic problems, in: *Proceedings of 9th IEEE International Conference on Fuzzy Systems* 2000.

Mee Hong Ling has a Bachelor (Hons.) degree in Computer and Mathematical studies from Oxford Brookes University, UK and a Master degree in Data Engineering (Computer Science) from Keele University, UK. She lectures in the Department of Computer Science and Networked Systems at Sunway University. She is currently working towards her PhD degree in Computing. Her research interests are in the areas of security, cognitive radio networks and reinforcement learning.

Kok-Lim Alvin Yau has a B. Eng. degree in Electrical and Electronics Engineering (First Class Honors) from the Universiti Teknologi Petronas, Malaysia (2005), a MSc (Electrical Engineering) from the National University of Singapore (2007), and a PhD (Network Engineering) from Victoria University of Wellington, New Zealand (2010). He was awarded the 2007 Professional Engineer Board of Singapore Gold Medal for being the best graduate of the MSc degree in 2006/07. He researches, lectures and consults in cognitive radio, wireless networking and applied artificial intelligence. He is a member of the Department of Computer Science and Networked Systems, Sunway University.

Junaid Qadir is an Assistant Professor at the School of Electrical Engineering and Computer Sciences, National University of Sciences and Technology, Pakistan. He completed his PhD from University of New South Wales, Australia in 2008 and his BS in Electrical Engineering from UET, Lahore, Pakistan in 2000. His research interests include the application of algorithmic, machine learning, and optimization techniques in networks. He serves as an Associate Editor for IEEE Access, and is the lead editor of the IEEE Access special section on AI-Enabled Networking to be published in Q3, 2015. He is a member of ACM, and a senior member of IEEE.

Geong Sen Poh has a Bachelor (Hons.) degree and a Master degree in Computer Science from Universiti Sains Malaysia, and a PhD degree in Information Security from University of London, UK. He is currently an Assistant Professor at University Malaysia of Computer Science & Engineering (UniMy), and a researcher at MIMOS Berhad. His main research interests include design and analysis of security protocols based on cryptographic schemes and digital watermarking. He has published in the field of watermarking and information security and is currently involved in various research projects. He also provides consultations on issues related to information security.

Qiang Ni received the B.Sc., M.Sc., and Ph.D. degrees from Huazhong University of Science and Technology, Wuhan, China, all in engineering. He is a Professor of Communications and Networking with School of Computing and Communications, Lancaster University, Lancaster, U.K. Prior to that, he led the Intelligent Wireless Communication Networking Group at Brunel University, London, UK. His main research interests are wireless communications and networking, in which he has published more than 100 papers. Prof Ni was an IEEE 802.11 Wireless Standard Working Group Voting Member and a contributor to the IEEE wireless standards.