4-2020

Machine Learning-Enabled Resource Allocation for Underlay Cognitive Radio Networks

Fatemeh Shah Mohammadi
fs2213@rit.edu

Rochester Institute of Technology

Machine Learning-Enabled Resource Allocation for Underlay Cognitive Radio Networks

by

Fatemeh Shah Mohammadi

Supervisor: Prof. Andres Kwasinski

A dissertation submitted in partial fulfillment for the degree of Doctor of Philosophy in Engineering

Engineering (PhD Program)

Kate Gleason College of Engineering

Computer Engineering Department

April 2020

Thesis Signature Form

Approvals

Dr. Andres Kwasinski

Dissertation Advisor	Signature/Date

Dr. Raymond Ptucha

Committee Member	Signature/Date

Dr. Drew Maywar

Committee Member	Signature/Date

Dr. Panos Markopoulos

Committee Member	Signature/Date

Dr. Nathan Cahill

Outside Representative	Signature/Date

Dr. Edward Hensel

PhD Program Director	Signature/Date

"There is no recipe, there is no one way to do things — there is only your way. And if you can recognize that in yourself and accept and appreciate that in others, you can make magic."

Ara Katz

Abstract

Due to the rapid growth of new wireless communication services and applications, much attention has been directed to frequency spectrum resources and the way they are regulated. Considering that the radio spectrum is a natural limited resource, supporting the ever increasing demands for higher capacity and higher data rates for diverse sets of users, services and applications is a challenging task which requires innovative technologies capable of providing new ways of efficiently exploiting the available radio spectrum. Consequently, dynamic spectrum access (DSA) has been proposed as a replacement for static spectrum allocation policies. The DSA is implemented in three modes including interweave, overlay and underlay mode [1].

The key enabling technology for DSA is cognitive radio (CR), which is among the core prominent technologies for the next generation of wireless communication systems. Unlike conventional radio which is restricted to only operate in designated spectrum bands, a CR has the capability to operate in different spectrum bands owing to its ability in sensing, understanding its wireless environment, learning from past experiences and proactively changing the transmission parameters as needed. These features for CR are provided by an intelligent software package called the cognitive engine (CE). In general, the CE manages radio resources to accomplish cognitive functionalities and allocates and adapts the radio resources to optimize the performance of the network. Cognitive functionality of the CE can be achieved by leveraging machine learning techniques. Therefore, this thesis explores the application of two machine learning techniques in enabling the cognition capability of CE. The two considered machine learning techniques are neural network-based supervised learning and reinforcement learning. Specifically, this thesis develops resource allocation algorithms that leverage the use of machine learning techniques to find the solution to the resource allocation problem for heterogeneous underlay cognitive radio networks (CRNs). The proposed algorithms are evaluated under extensive simulation runs.

The first resource allocation algorithm uses a neural network-based learning paradigm to present a fully autonomous and distributed underlay DSA scheme where each CR operates based on predicting its transmission effect on a primary network (PN). The scheme is based on a CE with an artificial neural network that predicts the adaptive modulation and coding configuration for the primary link nearest to a transmitting CR, without exchanging information between primary and secondary networks. By managing the effect of the secondary network (SN) on the primary network, the presented technique maintains the relative average throughput change in the primary network within a prescribed maximum value, while also finding transmit settings for the CRs that result in throughput as large as allowed by the primary network interference limit.

The second resource allocation algorithm uses reinforcement learning and aims at distributively maximizing the average quality of experience (QoE) across transmission of CRs with different types of traffic while satisfying a primary network interference constraint. To best satisfy the QoE requirements of the delay-sensitive type of traffics, a cross-layer resource allocation algorithm is derived and its performance is compared against a physical-layer algorithm in terms of meeting end-to-end traffic delay constraints. Moreover, to accelerate the learning performance of the presented algorithms, the idea of transfer learning is integrated. The philosophy behind transfer learning is to allow well-established and expert cognitive agents (i.e. base stations or mobile stations in the context of wireless communications) to teach newly activated and naive agents. Exchange of learned information is used to improve the learning performance of a distributed CR network. This thesis further identifies the best practices to transfer knowledge between CRs so as to reduce the communication overhead.

The investigations in this thesis propose a novel technique which is able to accurately predict the modulation scheme and channel coding rate used in a primary link without the need to exchange information between the two networks (e.g. access to feedback channels), while succeeding in the main goal of determining the transmit power of the CRs such that the interference they create remains below the maximum threshold that the primary network can sustain with minimal effect on the average throughput. The investigations in this thesis also provide a physical-layer as well as

a cross-layer machine learning-based algorithms to address the challenge of resource allocation in underlay cognitive radio networks, resulting in better learning performance and reduced communication overhead.

Acknowledgements

I would like to sincerely express my gratitude to my advisor Dr. Andres Kwasinski for guiding me during my Ph.D. program. His constant support and timely advice have provided me with a great opportunity to earn my Ph.D. degree. Dr. Kwasinski's insightful thoughts, excellent academic expertise and inspiring guidance have significantly helped me to achieve academic excellence. His diligent working attitude and pursuit of perfection have motivated me to pay attention to the details and achieve the best. His encouragement and support will continue to motivate me in the future. I am very grateful to him for being my Ph.D. advisor.

I want to thank Dr. Drew Meywar, Dr. Raymond Ptucha and Dr. Panos Markopoulos for their time and efforts in serving as my committee members. Their constructive and genuine advice has greatly helped me to improve this dissertation work.

I owe a lot to my parents. I am very grateful to them for their constant support and faithful love.

Contents

List of Figures

List of Tables

Abbreviations

ANN	**A**rtificial **N**eural **N**etwork
AMC	**A**daptive **M**odulation and **C**oding
ARQ	**A**uthomatic **R**epeat re-**Q**uest
BER	**B**it **E**rror **R**ate
CDMA	**C**ode **D**ivision **M**ultiple **A**ccess
CE	**C**ognitive **E**ngine
CNN	**C**onvolutional **N**eural **N**etworks
CR	**C**ognitive **R**adio
CRN	**C**ognitive **R**adio **N**etwork
CQI	**C**hannel **Q**uality **I**ndicator
DARPA	**D**efense **A**dvance **R**esearch **P**roject **A**gency
DSA	**D**ynamic **S**pectrum **A**ccess
DP	**D**ynamic **P**rogramming
DQL	**D**eep **Q**-**L**earning
FDMA	**F**requency **D**ivision **M**ultiple **A**ccess
FTP	**F**ile **T**ransfer **P**rotocol
HARQ	**H**ybrid **A**uthomatic **R**epeat **R**equest
IDS	**I**ntrusion **D**etection **S**ystem
ITU	**I**nternational **C**ommunication **U**nion
MCM	**M**onte **C**arlo **M**ethod
MDP	**M**arkov **D**ecision **P**rocess
MSE	**M**ean **S**quare **E**rror
MIMO	**M**ultiple **I**nput **M**ultiple **O**utput
MOS	**M**ean **O**pinion **S**core

ML	**M**achine **L**earning
NARX-NN	**N**on-linear **A**uto-**R**egressive exogenous **N**eural **N**etwork
RL	**R**einforctment **L**earning
SDMA	**S**pace **D**ivision **M**ultiple **A**ccess
PBS	**P**rimary **B**ase **S**tation
PU	**P**rimary **U**ser
PSNR	**P**eak **S**ignal **t**o **N**oise **R**atio
SBS	**S**rimary **B**ase **S**tation
SINR	**S**ignal **t**o **I**nterference **p**lus **N**oise **R**atio
SRN	**S**imple **R**ecurrent **N**etworks
SU	**S**econdary **U**ser
SVM	**S**upport **V**ector **M**achine
QoE	**Q**uality **o**f **E**xperience
QoS	**Q**uality **o**f **S**ervice
RL	**R**esource **A**llocation
TD	**T**emporal **D**ifference
TDMA	**T**ime **D**ivsion **M**ultiple **A**ccess
TDNN	**T**ime **D**elay **N**eural **N**etwor
VoD	**V**ideo **o**n **D**emand

*Dedicated to my beloved family for their support,
encouragement and love*

Chapter 1

Introduction

Radio spectrum is a limited natural source which is regulated around the world by international and national regulators. The International Communication Union (ITU) and in particular, its Radio communication Sector (ITU-R) have the core responsibility of the governance of the radio spectrum. At the international level of allocation which is done by ITU, the use of radio spectrum is regulated for different type of services on a regional basis. At the national level, in most countries the use of radio spectrum is managed by the government. For example, the Federal Communications Commissions (FCC) is responsible for radio spectrum regulation in the United States, while in the United Kingdom it is regulated by the Office of Communications (Ofcom). Radio spectrum allocation and management, traditionally, can be divided into two categories: licensed and unlicensed [5]. Over unlicensed bands, users can freely transmit without any licensing requirements, while in licensed bands, the right of the use of a frequency band is assigned through the sale of a license. This approach assigns fixed spectrum allocation, operating frequencies and bandwidths with constrains on power emission which limits their range. Hence, most communication systems are designed so that to increase spectrum efficiency within the assigned bandwidth using different modulation, coding, multiple antenna configuration and other techniques [6].

Since demands on wireless applications are explosively increasing, the radio spectrum is becoming more occupied. Because of this, the static spectrum allocation policy will result in spectrum scarcity. Conflict between the spectrum scarcity due to the traditional static spectrum allocation policy, and ever increasing service demands has motivated research groups to investigate innovative schemes, that can solve the problem of the spectrum allocation. However, radio spectrum usage pattern shows that large portion of the licensed bands are under-utilized. For instance, in [7] authors reported a study, conducted by the US Defense Advance Research Project Agency (DARPA), showing that only 2% of the licensed spectrum is utilized in USA. Authors in [8] also showed that only 22% of allocated spectrum is in use in urban areas, while this figure is less than 3% in rural areas. These findings led researches toward the definition of spectrum hole which is a band of frequencies already assigned to primary users (who are licensed to transmit on assigned frequency band), but at a particular time and/or specific geographical location, the band is not being utilized by the primary user [5]. Based on the evidences, which briefly mentioned here, the current fixed spectrum allocation policy is highly inefficient and results in under-utilized frequency bands. Consequently, a new communication paradigm is needed which leads to more efficient usage of the radio spectrum by exploiting the spectrum holes. In this new paradigm, unlicensed users or secondary users (SUs) will "opportunistically" operate in the unused licensed spectrum bands without interfering with licensed users or primary users (PUs), hence it enhances the efficiency of spectrum utilization. This model of spectrum sharing is called interweave dynamic spectrum access (DSA). There are two more models naming overlay DSA and underlay DSA which were later merged into the DSA spectrum sharing models.

1.1 DSA Models

As stated, there are three DSA models, including interweave, underlay, and overlay [1]. In interweave DSA model the SUs can have access to the spectrum only if the frequency band of interest is idle and PU is not active on the band. In this model PU has the absolute priority on the spectrum band and SU must vacant the band if PU wants to access the band. Since the SUs opportunistically utilize the spectrum holes in time, space and/or frequency domain, this model is also called opportunistic spectrum access. With the interweave DSA model, SU uses the cognitive radio to sense the surrounding spectrum environment, then selects one or more idle spectrum band(s), and switches to the selected band(s) to keep a seamless transmission. Hence, for this model SUs need to be equipped with a reliable spectrum sensing mechanism. In contrast to the interweave model, underlay model allows coexistence of primary and secondary users at the same time and over the same frequency band, provided that the accumulated interference from all SUs is less than the tolerable interference level of the PU. In this DSA model, power allocation for SUs is the first task needs to be taken into account.

The overlay DSA model, as more recent development of DSA, is similar to underlay DSA and allows concurrent coexistence of both users on the same frequency band but with a different constraint. The overlay DSA aims at maintaining the PU performance, as contrast to underlay DSA which constraints the interference from SUs to PU via limiting the SU's transmit power. Specifically, SUs are allowed to continue their transmission simultaneously with PUs over the same band as long as no performance degradation is guaranteed for PUs. There are two approaches to overlay DSA. The first approach is to use channel coding [1]. In this approach the packet which is being transmitted by the PU is known to SU, thus the SU transmitter can split its transmit power into two parts to transmit its own (SU) packet along with the PU packet to enhance the total power received at the PU receiver, such that the PU received signal to interference plus noise ratio (SINR) does not degrade. To cancel the the interference to the SU caused by transmitting the PU packet, SU transmitter can use the dirty paper coding to precode the SU packet. Another approach for the overlay DSA model is to use network coding [9]. In this approach, an SU serves as a relay node between PU nodes. While relaying a PU packet, the SU may encode its packet onto the PU packet through network coding. Therefore, transmission of the SU packet does not need separate spectrum access, while the PU performance does not degrade as well.

Regarding the network spectrum sharing control architectures, there are two of them in CR networks which are networks with centralized control and networks with distributed control. They can be further categorized according to the spectrum allocation behavior as Cooperative and Non-cooperative. In the following we briefly explain each paradigm.

1.2 Centralized and Distributed CR networks

In the centralized network setting, resource allocation decisions are taken by a secondary base station or a central spectrum server. In this architecture, the base station gathers the information from the SUs or a dedicated sensor network. Using this information it runs an assignment algorithm and informs the SUs of the resources to be utilized. In this version of network setting an assignment of predefined control channel is needed. Having information on the global setting, the base station can provide a globally optimal resource assignment. However, there are several problems regarding to this architecture: requirement of a common control channel and high processing complexity at the base station. In this type of networks common control channel should be pre-assigned and all SUs should have interference free access to it. Distributed networks are preferred to centralized when there is no preexisting infrastructure available or when the centralized networks are not scalable. CRs use distributed algorithms while performing resource assignment in the decentralized setting. Each user does its own sensing equipments and coordinates with the nearby cognitive radios and determines who gets which resource. These algorithms are iterative so the neighborhood to which the CR sends data and from which it receives data should be limited. Otherwise it can incur large delays and the network will become unscalable. The distributed networks are also not completely immune to the problem of common control channel because to communicate with one another, transmitter and receiver should perform a handshake and it should happen on a channel both transmitter and the receiver know about.

1.3 Cooperative and Non-cooperative CR Networks

All the centralized CR networks are cooperative networks in principal and some distributed networks can be cooperative as well. The choice of a utility function plays a key part in defining a network as cooperative and non-cooperative. Each user tries to maximize the individual gain it can achieve without considering about the utility of others in non-cooperative networks.

In contrast to that a user in a cooperative network tries to maximize its own utility while satisfying the minimum utility required by other users in the network. One advantage of non-cooperative networks over the cooperative ones stems from the advantage of being able to function with minimum communication with other CR users. Cooperative networks can improve the overall network utilization with the burden of higher communication overhead. In situations where the benefits of cooperation surpasses the cost of communication one should choose a cooperative solution.

1.4 Motivation

Cognitive radio (CR) is the enabling technology for DSA, [10]. Unlike a conventional radio which can only operate on a predefined spectrum band based on regulatory restrictions, a cognitive radio is capable of operating in different spectrum bands. In order to be able to operate in different bands, CR must have the ability to sense and understand its wireless environment and proactively change its mode of operation as needed in order not to interfere with PU transmission. These set of fundamental activities, which is called cognition cycle [11], is performed by a CR in order to satisfy the end-user needs. The set of activities are as follows: observing the environment and acquisition of information such as spectrum occupancy or interference temperature, orienting itself, learning from past experiences, decision making and finally performing actions.

A cognition cycle by which cognitive radio interacts with the environment is illustrated in Fig. 1.1. According to this interpretation, the action phase consists of (re)configuring the CR to provide enhanced communication quality with respect to end-user goals. Such configuration can be for example the choice of the wireless radio interface to be used for communication, or the tuning of the communication system's parameters; the observation phase collects statistics from the device which characterize the external environment, such as traffic load patterns, SNR measurements, packet error rate, round trip time, etc.; the orientation phase consists of understanding the impact on communication performance of the external environment and of possible system configurations. This is achieved by identifying a functional relation between measurements and configuration parameters and

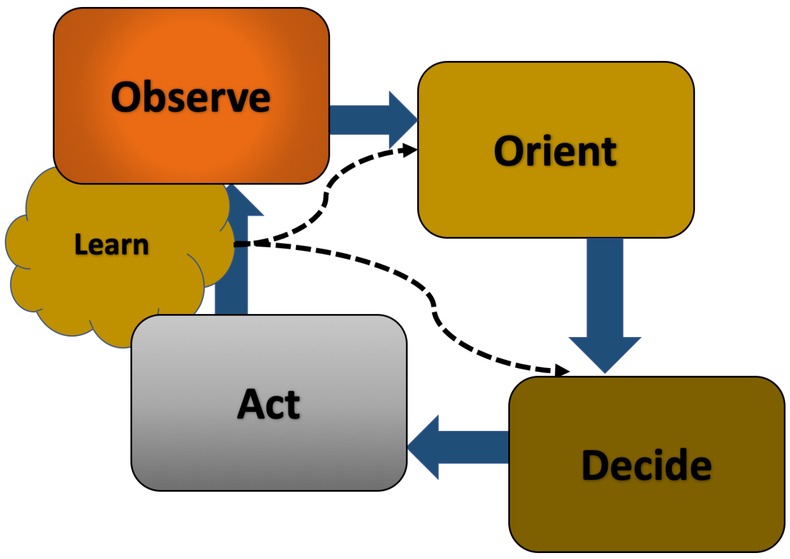

FIGURE 1.1: The cognitive radio's cognition cycle.

different aspects of communication performance (e.g, throughput, delay, reliability); the decision phase is the solution of the performance optimization problem, i.e., it is a search in the space of possible configurations which aims at finding the one that best satisfies user-defined goals, which are expressed in terms of high-level performance metrics such as application-layer throughput, delay and reliability, as well as cost, power consumption, etc; and finally, the learning phase consist of evaluating the outcome of the decisions which have been made, thereby gathering knowledge to be exploited in future orientation phases with the aim of being more effective in the decision phase.

The brain of a CR where the set of activities associated with the cognitive cycle is implemented is called Cognitive Engine (CE). Machine learning (ML) can be widely used in implementing the learning, orientation and decision phases of CRs (and specifically CEs). We explore in this thesis the application of two machine learning techniques including neural-network based supervised learning and reinforcement learning in this context. There are various reinforcement learning techniques among which we consider the most common form, widely considered to be Q-learning, in both standard table-based and deep Q-learning, and deep Q-learning. We focus on underlay cognitive radio networks where we need to devise resource allocation algorithm for the SUs such that it ensures the non-interfering co-existence of primary and secondary networks. In particular, we use machine learning technique to allocate for the SUs radio resources (transmit power and rate in this work) such that not only the PU interference constraint is met, but also the network performance metric in SN is maximized.

1.5 Why ML is needed?

Recent advances in technology, the ever increasing computing power of machines and the instant connectivity provided by the Internet anywhere and at anytime in the world, has enabled companies to capture trillions of bytes of information everyday, in what is known as Big Data. Such information comes from the billions of sensors connected to computers, automobiles, mobile phones, home appliances and in general everyday objects with ability to sense, create and communicate data. As such, this massive amount of data generated and collected every second motivated companies to explore that data using data analytic methods, in order to create better solutions that would benefit them while also please their customers [12].

Regarding the wireless communications, rapid development of intelligent devices (such as smart phones, cars and home) and development of cloud computing and network virtualization has led to exponential increase in data traffic. On the other hand, the networks have become more complex as they involve a multitude of devices, drive different protocols and support diverse applications. In wireless networks, for instance, different types of cells (each associated with a specific coverage such as macro-cells, pico-cells and femto-cells) with different transmission power range, working mechanism and communication technologies (such as ZigBee, WiMAX, Bluetooth and LTE) have been applied and have led to more challenges in effectively managing and optimizing the network resources. Inserting more intelligence into networks is one possible solution to these challenges. This intelligence can be brought by deployment of ML and cognitive techniques which always favor from the collected massive amount of data. In fact, ML algorithms rely on collecting and analysing data in order to find patterns and relationships between them and produce a model that can relate the input to the output, instead of trying to develop a complex and complete model of the system. As a result, these algorithms are able to learn, reason and make decisions without human intervention [13].

In addition, another clear advantage of ML is that these algorithms are able to generalise. For example, considering the task of mobile network optimization, if a model had to be developed for every possible situation of the network, with users in all possible positions and all base stations with different power, interference and load levels, it would be impossible. As such, solutions that analyse data and are able to create a model based on their observations are much more feasible. This occurs because ML solutions are able to learn from data and make predictions if new unseen data is fed into the model, being much more general than analytical approaches, as they do not require the entire model to be trained again or rebuilt from zero [13, 14].

Furthermore, another key advantage of ML algorithms is when dealing with complex tasks. Similar to the case above, in which for some applications it is impossible to create an analytical model for every possible solution, in very complex domains, traditional approaches would also not work. For example, considering the task of teaching a self-driving car how to drive. The car has a large number of sensors, inputs, images from different cameras and also lots of possible actions, such as accelerating, braking, turning the wheel, changing gears, etc.. As it can be seen from this example, when tasks are extremely complicated and have many variables and parameters, traditional analytical approaches or controller design solutions are not very suitable, as the solutions required to solve these problems would be too complex and costly. As such, in those cases ML solutions also excel at, as they are able to learn from the great amount of data gathered and generated from these complex applications and determine the best action [13, 15]. ML algorithms, and more recently deep learning, have shown comparable human performance in certain tasks, such as image classification or playing certain games like chess, Go, backgammon, and video games [16, 17]. As such, with the constant development of more robust and powerful computers and algorithms, the possibilities of what these intelligent algorithms can do are practically unimaginable.

1.6 ML as an Enabler of CR

ML algorithms allow the system to learn the structural patterns and models from training data. A ML approach usually includes two main phases: a training phase and a decision making phase. During the training phase, the system model is learnt using the training data set. Whereas, during the decision making phase the trained model is used for each new input to estimate the output of the system. As stated earlier, ML techniques bring intelligence to cognitive radio networks by performing data analysis and network optimization. Such intelligence derived from learning capability of ML enables CRs to autonomously learn to make optimal decisions to adapt to the network environment. Before addressing the existing learning effort to address issues in CRNs, we briefly explore the main ML algorithms.

1.6.1 ML Algorithms

ML algorithms are basically divided into three categories, as follow:

- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

Below a brief description of each category is presented. For more insightful discussion on ML theory and its concepts, the works [18, 19] can be referred.

1.6.1.1 Supervised Learning

Supervised learning, as the name implies, requires a supervisor (teacher) to supervise (train) the system. For training, it requires a data set which has information about both input and output data. For every input data, the teacher knows the answers (output) by which the teacher can correct the predictions as the algorithm iteratively makes them. More formally, supervised learning is an algorithm that learns the underlying model in data (the relation between input and output data) that best represents the data, and is able to make predictions for newly, unseen data points. In addition, supervised learning algorithms are further split into two main categories, depending on type of their output variable. If the output variable is discrete value, such as spectrum band being busy or idle, the supervised learning problem is referred to as a classification problem. While, if the output variable is a real or a continuous value, such as the throughput in primary network, the supervised problem is considered as a regression problem. Supervised learning algorithms range from simple ones, such as linear regression, logistic regression, k-Nearest Neighbors and decision trees, to more complex ones, such as, support vector machines (SVM), neural networks, and its variant (such as convolutional neural networks (CNN) and deep neural networks)[14].

Neural Network

Artificial neural network (ANN or simply NN) is a ML technique and inspired by the human brain functionality. In fact, artificial neural networks are made up of artificial neurons (as emulation of biological neurons) interconnected with each other to form a programming structure to mimic organisation and learning procedure of biological neurons. Due to the special ability of human brain in parallel data processing (in massive volume), it performs many tasks much faster than fastest computers. NNs are meant to mimic such substantial performance, specially for the problems with cognitive or associative tinge. To this end, NNs have been applied to time-series prediction, pattern (image/speech) recognition, regression and function approximation, classification and adaptive control.

NNs, as stated, consists of pools of simple processing units called neurons. The neurons are categorized into three groups: input neurons (which organize the input layer) receive input from the outside; output neurons send the data out of NN and form the output layer; and hidden neurons (which comprise the hidden layer) receive data from within the NN and output the data to another layer inside the NN. Fig. 1.2 illustrates the typical structure of NN. Each neurons sends its signal over large number of weighted connections. Each connection is defined by a w_{jk} associated with which shows the effect that signal from neuron j has on neuron k. The output of neuron k is first the summation over the weighted version of all neurons in the prior layer as well as bias offset, and then activation the result of

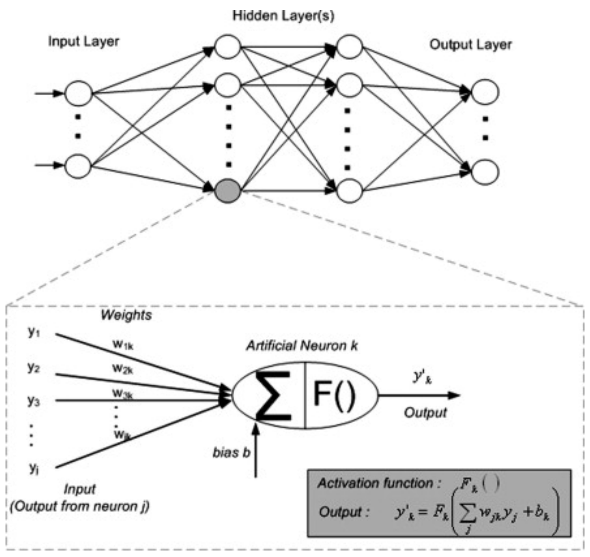

FIGURE 1.2: A Basic Neural Network.

summation using a activation function F_k, as follow [20]:

$$F_k(\sum_{j=1} B w_{jk} y_j + b_k), \tag{1.1}$$

where B denotes the number of neurons in the previous layer. The activation function is usually sort of threshold function, such as rectified linear unit function, sigmoid function and hyperbolic tangent sigmoid.

NN is itself divided into two categories of feed-forward NN and recurrent NN. In feed-forward NNs, the data enters at the inputs and is forwarded through the network layer by layer, until it arrives at the outputs. While recurrent NNs also contain feedback connections, which are connections extending from outputs of neurons to inputs of neurons within the same or previous layers. Examples of recurrent networks have been presented in [21].

1.6.1.2 Unsupervised Learning

On the other hand, unsupervised learning algorithms are useful when the data set only contains the input data and does not have any ground truth labels [13, 22]. As such, these algorithms do not have a supervisor (i.e. output data) to lead the training procedure. However, they search for similarity in the data and form groups of similar examples (known as clusters

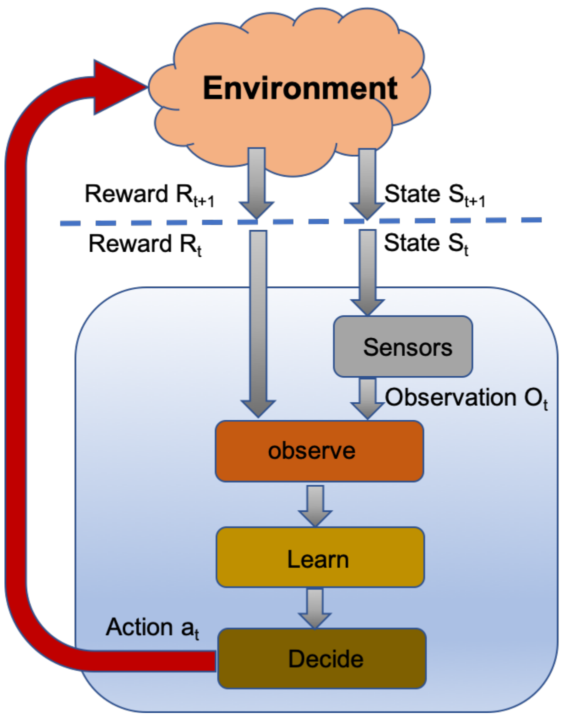

FIGURE 1.3: The reinforcement learning cycle.

too). Since unsupervised learning tries to estimate a model without having access to labeled output data, unsupervised learning algorithms are mainly grouping algorithms (clustering), such as K-Means and mixture models [13, 22].

1.6.1.3 Reinforcement Learning

As mentioned, in supervised learning the relational model between input and output is developed using an external supervisor. But reinforcement learning (RL) is quite different than supervised learning, as it deals with interactive problem in which generating examples of desired behaviour are quite hard or even impractical to achieve [23]. In fact, RL is adequate for learning dynamic models, and specifically in interactive or unknown situations where an agent has to learn from its own experience of interaction with the environment, RL is the most appropriate solution [23]. RL is a ML technique with an assigned goal meant to be achieve. In RL the agent interacts with the surrounding environment by taking actions and receiving rewards that show the effect of the action on the environment. After taking an action the entire environment (system) transitions into a new situation (known as state) (as shown in Fig 1.3) [23, 24]. At the beginning of each learning cycle, the agent receives a full or partial observation of the current state, as well as the accrued reward. By using the state observation and the reward value, the agent updates its policy (e.g. updating the Q-values) during the learning stage. Finally, during the decision stage, the agent selects a certain action according to the updated policy. The agent receives a reward only if the taken action results in transition toward the optimal state, otherwise it is penalized (meaning it receives a negative or small reward). Basically, the agent and environment interact continuously at certain time-steps. At each time-step, t, the agent receives a representation of the environment's state and selects an action according to a policy (π). On the next time-step, $t + 1$, as a consequence of its action, the agent receives a reward (r_{t+1}) and transitions in a new state. The goal of an agent is to maximize its total cumulative reward. Based on this, RL algorithm is divided into four main components [23]:

- Policy: dictates the behaviour of the agent at each state s, basically it determines which action a is chosen at state s.
- Reward (r_t): a numerical value given by the environment to the agent as the immediate return of taking an action, and that the agent tries to maximize over time.
- Value function (Q function or action-value function): indicates the expected value of visiting a state, $Q(s, a)$, or the value of taking an action in a specific state.
- Environment: includes everything outside the agent.

Regarding the RL algorithm, there is term know as exploration-exploitation which determines how new action is selected. If the new action at state s is selected based on the developed policy at that state, the agent exploit the best action. While if the new action is selected randomly, the agent performs exploration, meaning that the agent explores new actions, in order to determine if there are possible actions that lead to a better cumulative reward. Furthermore, RL algorithms can be divided into three main categories, as follow:

- Dynamic programming (DP): in which the agent has a perfect model of the environment, given by a Markov decision process (MDP), and the goal is to learn the optimal policy (in order to choose the best actions).
- Monte Carlo methods (MCM): in this case, it is not assumed that there is complete knowledge about the environment. Thus, the agent must learn either online, by experiencing the environment, or through simulated experiences, in which the environment is represented by a very simple model.
- Temporal-difference learning (TD learning): which can be defined as a combination of MCM and DP. Just like MCM, TD learning agents can learn directly from their experience with the environment, without the need of the complete environment dynamics. Furthermore, similar to DP, TD algorithms update their estimates (either a policy or a value function) based on other learned estimates.

These algorithms can be further divided into On-Policy or Off-Policy, depending on how learning is performed [23]:

- On-policy learning: the agent updates its value function and estimates the return (the total discounted future reward) assuming that the current policy continues to be followed. The performance of these algorithms is evaluated via on-policy interactions with the environment. Specifically, it is an algorithm that, during training, chooses actions using a policy that is derived from the current estimate of the optimal policy, while the updates are also based on the current estimate of the optimal policy.
- Off-policy learning: the agent updates its value function and estimates the return assuming a different policy than the one that is being followed. off-policy learning is good at predicting movement in robotics. Off-policy learning can be very cost-effective when it comes to deployment in real-world reinforcement learning scenarios. The characteristic of the agent to explore and find new ways and cater for the future rewards task makes it a suitable candidate for flexible operations. Imagine a robotic arm that has been tasked to paint

something other than what it is trained on. Physical systems need such flexibility to be smart and reliable.

With regard to RL there are also two famous terms, denoted as exploration and exploitation. Given that the agent still does not know the optimal policy, it often behaves sub-optimally. During training, the agent faces a dilemma: the exploration or exploitation dilemma. Exploration is the selection and execution (in the environment) of an action that is likely not optimal (according to the knowledge of the agent) and exploitation is the selection and execution of an action that is optimal according to the agent's knowledge (that is, according to the agent's current best estimate of the optimal policy). During the training phase, the agent needs to explore and exploit: the exploration is required to discover more about the optimal strategy, but the exploitation is also required to know even more about the already visited and partially known states of the environment. During the learning phase, the agent thus can't just exploit the already visited states, but it also needs to explore possibly unvisited states. To explore possibly unvisited states, the agent often needs to perform a sub-optimal action. With that being said we can discuss one of the most commonly used methods in RL is the ϵ-greedy, which states that with a probability $p = (1 - \epsilon)$ the action that results in the maximum value known by the agent is chosen, whereas with probability $p = \epsilon$ an action is chosen at random. Furthermore, when a decaying ϵ is chosen, this policy exhibits a nice trade-off in terms of exploration and exploitation, in which in the beginning, because the ϵ is quite larger, the agent will favour exploring new actions, while later, due to the decaying ϵ rate, the agent favours the exploitation of the best actions [23].

Q-Learning

One of the most popular algorithms in RL is Q-Learning. First proposed by Watkins, in [25], Q-Learning is a TD Learning method that learns an action-value function, $Q(s; a)$, which represents the expected value of an agent being in a certain state and taking a specific action. Q-Learning is a method that, at each step at a state s_t, chooses an action that maximizes its value function. This function, $Q(s_t; a_t)$, indicates how good is taking an action at a specific state according to an immediate reward r. The agent repeatedly makes decisions and finally obtains its optimal policy which maximizes the expected sum of discounted reward:

$$Q(s,\pi) = \textstyle\sum_{t=0}^{\infty} \gamma^t . E(r_t|\pi, s_0 = s), \tag{1.2}$$

where π is the local strategy and $s_0 = s$ is the initial state. According to Bellman's principle of optimality [26], the solution to (1.2) can be obtained by taking the optimal action if all the strategies thereafter are optimal:

$$Q^*(s,\pi^*) = \max_a [r(s,a) + \gamma \sum_{s'} p(s'|s,a) Q(s',\pi^*)]. \tag{1.3}$$

Further, $Q^*(s,\pi^*)$ in (1.3) can be approached by the Q-function, which is updated as follows [23, 25]:

$$Q_{t+1}(s_t,a_t) = (1-\alpha_t) Q^t(s_t,a_t) + \alpha_t [r_t(s_t,a_t) + \gamma . \max_a Q_t(a, s_{t+1})], \tag{1.4}$$

where $Q(s_t;a_t)$ is the current action-value function, α is the learning rate, r_t is the immediate reward, γ is the discount factor and $\max_a Q_t(a, s_{t+1})$ is an estimate of the optimal future action-value function at the next time step, over all possible actions, a. In (1.4), the back-up, which is defined as what the algorithm stores in memory, is represented by the right side of the equation.

Deep Q-Learning

Equation (1.2) is a value iteration algorithm that converges to the optimal action-value function if $t \to \infty$. This approach is impractical. Consequently, a function estimator is used to estimate the optimal action-value function. In a Deep Q-Learning (DQL), a neural network has been proposed as an efficient nonlinear approximator to estimate action-value function $Q_i(s,a;\theta) \approx Q^*(s,a)$, [16]. Deep Q-learning is suitable for the problems where the dimension of state-action space (possibly continuous) is high. In this thesis we used a fully connected feed-forward Multilayer Perceptronron (MLP) network to approximate the action-value function.

DQL takes advantage of neural network as a powerful non-linear function approximator to estimate the action-value function. To improve the learning performance, we include a technique known as "experience replay" to the DQL algorithm. In experience replay, at each time step, the experience of each agent with the environment is stored as a tuple $e_i(t) = (a_i(t), s_i(t), r_i(t), s_i(t+1))$ into a replay memory. The replay memory of the agent, denoted as $D(t) = e(1), \ldots, e(t)$, and its entries is used to update the parameters of the action-value function. Basically, at each time step random mini-batch of entries in memory is selected to train the parameters θ of the action-value function. This approach has several advantages over standard Q-learning. First, each step of experience is potentially used in many weight updates, which allows for greater data efficiency. Second, learning directly from consecutive samples is inefficient, due to the strong correlations between the samples; randomizing the samples breaks these correlations and therefore reduces the variance of the updates. Third, when learning on-policy the current parameters determine the next data sample which later are used to train the next parameters. This will lead to see how unwanted feedback loops may arise and the parameters could get stuck in a poor local minimum, or even diverge [23]. By using experience replay the behavior distribution is averaged over many of its previous states, smoothing out learning and avoiding oscillations or divergence in the parameters.

1.6.2 Learning Applications in CRNs

In the following, we review the existing learning applications to address issues in CRNs. These efforts can be generally divided into five categories of:

- Traffic Classification
- Routing Optimization
- Quality of Service (QoS)/ Quality of Experience (QoE) Prediction
- Resource Management
- Security

Below a brief description of each category is presented.

Traffic Classification

Traffic classification is identification of different traffic flow types in the network. Such information provides a network operator with a way to perform a fine-grained network management. In fact, the network operators with the help of traffic classification can more efficiently handle the various services and allocate different resources. ML-based traffic classification can be itself grouped into two approaches: application-aware traffic classification and QoS-aware traffic classification. In application-aware traffic classification the applications of traffic flow are identified [27–30]. Such classification usually involves three major tasks. First, collecting flows of different type of traffic, sorting them and extracting flow statistics and ground-truth training data from end devices and access devices; second deciding about which classifier to use to identify the name of the application such as Facebook, YouTube and etc., finally the flow type (such as video content, audio file, file sharing and Instant Messaging (IM)) can be even classified in case if a finer-grained traffic classifier is needed. However, due to the dramatic growth of applications on the Internet, it is difficult and also impractical to identify all applications. Since many different applications may belong to a QoS class, the applications can be classified according to their QoS requirements into different QoS classes. The QoS requirements can be delay, jitter or loss rate. This is what QoS-aware traffic classification approaches aim to do: categorizing (putting) traffic flows into different QoS classes [31]. The performance of learning algorithms used to implement traffic classification (which is mainly supervised learning algorithms) tightly depends on training data set, its dimension and volume.

Routing Optimization

Routing is a fundamental network function as it controls the traffic flows of the transmission links. Inefficient routing decisions may lead to over-loading of network links and consequently end-to-end transmission delay increase. Thus, traffic flows routing optimization is an significant research problem.

The two widely used algorithms in this field include short path first (SPF) algorithm and heuristic algorithms [32]. However, shortcomings associated with these algorithms (such as high computational complexity and resultant inefficient use of network resources) paved the way for development of ML-based routing policies. Ml-based routing algorithms can quickly give near-optimal solutions, of course after they were trained. Since the routing optimization problem can be seen as a decision-making task, RL-based algorithms have been applied significantly in this context. In particular, the network has seen as the environment, composed of the traffic states and the network have been usually considered as the state space and the reward has been defined based on the metrics targeted to be optimized [33–35]. Note that traffic prediction can enhance the performance of routing optimization as it reduces the transmission delay by modifying the switches' flow tables in advance [36]

QoS/QoE Prediction

ML-based QoS prediction approaches aim at discovering the quantitative correlation between the network key performance indicators (KPI) (such as packet size, transmission rate and the length of the queue) and QoS parameters to improve the QoS management. For example, [37] studies the problem of estimation of the network delay given the traffic load and the overlay routing policy. Authors in [38] limited their study to the specific application and performed application-aware QoS prediction for video on demand (VoD) application. As the QoS parameters (e.g. delay, throughput, jitter and loss rate) are often continuous variables (data), the QoS prediction problem can be seen as a regression task; as a consequence the supervised learning is an efficient learning paradigm to apply in this regard.

To have an end-user centric assessment of the quality of a received service, QoE subjective metrics are often used. Mean opinion score (MOS) is the most widely used QoE metric [39] which gives QoE values in five levels being as excellent, good, fair, poor and bad. Such values are obtained using subjective methods which necessitate availability of number of users to rate the quality of a specific service. Since the subjective methods are time consuming and expensive in cost, the QoS parameters are used to objectively model the QoE values. ML is an effective method which is used to learn the relationship between QoS parameters and the QoE values. Since the QoE values (MOS values) are discrete data, QoE prediction lie under the realm of classification, and accordingly supervised learning algorithms are the best fit for this task [40].

Resource Management

To improve the network performance an efficient network resource management is the primary requirement needed to be met by network operators. There are mainly three type of resources need to be managed efficiently including catching, networking and computing resources. Networking resources are used to deliver the data through the network and includes spectrum and transmit power, to name a few. Dynamic spectrum access (DSA) has accounted large body of research under this resource management category.Maximising (satisfying) the network QoS/QoE (requirements) is usually the background motivation for optimizing the management of these type of resources. On the other hand, catching resources are used to store the frequently requested data at devices to reduce the data transmission delay, resulting in improvement of the network performance [41]. Development of applications requiring more computational capability such as face recognition and augmented reality as well as limited computing resources and battery capacity motivated the researchers to consider more efficient usage of computing resources. For example, to offload the computing tasks, edge computing technologies are used to deploy the computing resources closer to the end-users [42, 43]. The work in [44] focused on jointly allocating the networking, caching and computing resources. In particular, it formulated the resource allocation problem as a joint optimization problem by considering the gains of networking, caching and computing and deployed deep Q-learning algorithm to obtain the best resource allocation policy for the smart cities. [45] explores ML-based spectrum sharing solution for a system where LTE and WiFi coexist together and share the same unlicensed spectrum resources while each has its own separate controllers. We explore more studies in next chapters that focused on resource allocation management for the specific case of spectrum sharing problem.

Security

Since the secure networks can be only accepted and used by users, security has been always the vital aspect in development of any network which needs to be considered by the network operators. Intrusion detection system (IDS), as an important network security element, is responsible to monitor the events in the network and identify the possible attacks. According to how IDSs identify the network attacks, they are categorized into two types; signature-based and anomaly-based IDS [46]. When traffic flows arrive, signature-based IDS compare these traffics with known signature and if they are not matched, the traffic flows are classified as malicious activities. However, there are some shortcomings associated with signature-based IDS. As in such IDSs humans are responsible for creating the signature of unknown attacks, they can only identify attacks whose signature is available, while the development of new attacks makes the signature update difficult. The long time associated with comparing all signature is the next main disadvantages of signature-based IDS.

Compared to the signature-based IDS, anomaly-based IDS performs statistical analysis on the collected data related to the behavior of legitimate users to be later used to develop a model. The way it works is that each arriving flow is compared with the model and if its behavior has a significant deviation from the model, it is marked as anomaly. Thus, anomaly-based are able to detect new type of attacks. ML methods have been widely used in anomaly-based IDS. As stated anomaly-based IDS need training data set to develop the model, thus, such intrusion detection can be considered as a classification task (performed by supervised learning algorithms). As a consequence, the high dimensional of input data set has impact on the ML-based intrusion detection. to speed up the detection process while maintain the accuracy, feature reduction is used to reduce the input data dimensionality. Two well-known and effective methods in this regards are feature extraction and feature selection. In feature extraction a set of existing features are used to extract one new feature, which results in reduction in dimensionality. In feature selection a subset of appropriate features are selected (instead of using all features). The works [47, 48] studied the ML-based anomaly detection problem.

1.7 Thesis Outline

This thesis consists of two core chapters where we present our machine learning based resource allocation algorithms for underlay cognitive radio networks. Chapter two discusses the supervised learning-based algorithm where a recurrent neural network is used at each SU to predict the effect of its transmission on the nearest primary link. Specifically, we present an underlay DSA technique to infer, without tapping into feedback or control channels from another network, the modulation order and the channel coding rate (and accordingly the effective bit rate) used in the transmission of the primary network, and to leverage this inference in the realization of a fully autonomous and distributed underlay DSA scheme. This ability to estimate the effective throughput enables a finer control knob for a more accurate power allocation with less harmful effect on PN transmissions.

In the third chapter, we explore the application of reinforcement learning and its two main variant in allocating the resources for underlay DSA. As opposed to the second chapter, where we consider a system with more than one primary link, considered system in this chapter consisting of a single primary link. This chapter itself consists of three sub-sections; in first section we develop a physical-layer resource allocation algorithm while in the second section a cross-layer algorithm is developed to incorporate into our design the parameters from higher layers of protocol stack. This is because in measuring the QoE for real-time video streaming SUs, not only the PSNR of the received video traffic but also its experienced end-to-end delivery delay play a key role. To guarantee the end-to-end delivery delay for the video traffic we monitor the state of the queue at the data link layer (where the video packets are buffered to be later transmitted) and incorporate the result of this monitor into our machine learning model. In the third section we explore the solution of transfer learning to address the problem of long learning time associated with reinforcement learning. This is because reinforcement learning devises the best resource allocation decision (policy) only after performing a numerous trials and by rewarding the successful decisions and penalising the failed ones. The philosophy behind the transfer learning is to allow well-established and expert cognitive agents (i.e. base stations or mobile stations in wireless communication context) to teach newly activated and naive agents. This exchange of learning information is used to improve the performance of a distributed cognitive radio network. With regard to transfer learning there are three main challenges: who to obtain the knowledge, the level of information exchange and when to stop the information exchange. We, specifically, explore the first two challenges regarding the transfer learning and their effect on the learning performance of the RL algorithm. In the last chapter we conclude the thesis.

Chapter 2

Fully Autonomous Supervised Learning-Based Resource Allocation Framework for Underlay Cognitive Radio Networks

In this chapter we develop a supervised learning-based CE for the SUs operating under the underlay DSA scenario. The neural network is used as the machine learning model to implement the supervised learning. As stated in the first chapter, the data set is used as a teacher to supervise the learning agent (and specifically the neural network) to detect the relation between the inputs (coming from the environment) and output. The deployed neural network in the work being presented in this chapter will be trained using a data set collected from a network simulation with a set up which will be described afterwards.

The motivation for developing this work is to answer the two key challenges in underlay DSA, which are how to establish an interference limit from the primary network (PN) and how cognitive radios (CRs) in the secondary network (SN) become aware of the interference they create on the PN, especially when there is no exchange of information between the two networks. These challenges are addressed in this chapter by presenting a fully autonomous and distributed underlay DSA scheme where each

CR operates based on predicting its transmission effect on the PN. The scheme is based on a cognitive engine with an artificial neural network that predicts, without exchanging information between the networks, the adaptive modulation and coding configuration for the primary link nearest to a transmitting CR. By managing the effect of the SN on the PN, the presented technique maintains the relative average throughput change in the PN within a prescribed maximum value, while also finding transmit settings for the CRs that result in throughput as large as allowed by the PN interference limit. It is shown through simulation results that the ability of the cognitive engine to estimate the effect of a CR transmission on the full adaptive modulation and coding (AMC) mode that is used at a PN link translates into a much more fine underlay transmit power control and increase of the CR transmission opportunities, compared to a scheme that can only estimate the modulation scheme used at the PN link.

2.1 Literature Review

Despite having been studied for almost two decades, the main challenges in underlay DSA remain on how to establish the interference threshold for the PN links and how the SUs autonomously become aware of the interference they create on the PN, specially when following an ideal operating setup where there is no exchange of information between primary and secondary networks. In order for the SN to assess its effect on the PN and protect the PN transmissions, researchers have proposed different techniques that usually assume that the secondary transmitter (SU-TX) knows the gain of the primary channel (that between the primary transmitter, PU-TX, and the primary receiver, PU-RX) and/or the cross-channel gain from the SU-TX to the PU-RX, [49–56]. The common theme between these works is that they make use of the information that is sent over a feedback channel from the PU-RX. Since feedback channels are part of most wireless communication standards [57, 58], they have often been used, under the assumption that SUs can access them, to not only estimate the primary channel gain, but also assess the effect of the SN on PN transmissions. Examples of a CR obtaining information about the primary link from a PN feedback channel are found in [49–52, 56] for the case of the rate/power control feedback channel, in [53] and [54] for the ARQ feedback channel, and in [55] for the feedback of the channel state information (CSI). In general, by relying on listening to feedback channels from the PN all works mentioned above share a setup where primary and secondary networks are not completely separated and exchange information with each other. In fact, the access of a control channel from another network calls into question as to whether there are really two separate network or, as we would argue, a single network with two different types of nodes.

With a different approach, the works in [59] and [60] proposed solutions to obtain the cross-channel gain without listening to the PN feedback channels. The typical process in these works consists of the SU observing a change in the primary's waveform power and/or modulation order that results from the transmission of a probe message from the SU, [59], or the SU acting as a relay by sending the amplified version of the signal received from the PU, [60]. However, in these works a CR transmitter is not able to estimate the cross-channel gain and later assess the effect of SN on PN, unless it observes a change in primary signal power and/or modulation order. Moreover, these work did not considered the combined effect from scenarios with multiple links in the PN and the SN, as they focused on a setup with one link in each network. In contrast to these works, the technique to be presented here is not limited by the need to observe a change in transmit power or modulation order and is able to directly estimate the effect of an SN transmission on the PN much more accurately by also estimating the channel coding rate in a PN link. Moreover, the work herein is on scenarios consisting on multiple links in the PN and SN. In addition, our presented technique meets a key requirement by not relying on any information exchange between the networks but, instead, takes advantage of the use at the PN of adaptive modulation and coding (AMC), a technique where the modulation scheme and channel coding rate, a pair of parameters known as the AMC mode, are adapted based on the quality of the transmission link. The use of AMC has been part of all high performance wireless communications standards developed over the past two decades and, thus, expected to be used by a typical PN [61–63]. By estimating the AMC mode used in a primary link, it becomes possible for a CR to learn the signal-to-interference-plus-noise ratio (SINR) experienced at that link because the AMC mode in use depends on the link's SINR.

Therefore, in this work we propose an underlay DSA technique that configures the transmit power for a SU based on estimating the throughput at its nearest PN link that corresponds to the AMC mode that would be chosen based on the interference created by the SU's transmission. The proposed technique works in a distributed way and is not limited by the number of links in the PN or the SN. The core of our technique is a non-linear autoregressive exogenous neural network (NARX-NN)-based cognitive engine that estimates the throughput on a PN link (equivalently, the full AMC mode), an indicator of the interference induced by the SU on the PN link, using as input an estimate of the modulation scheme being used at the PN link. As such, to the best of our knowledge, our presented cognitive engine is the first to be capable of estimating the channel coding rate setting for an AMC mode, a capability well beyond the estimation of the modulation scheme that has been realized through multiple signal processing or machine learning techniques. While modulation classification, the technique to estimate the modulation scheme used in a radio waveform, is applied in our DSA technique to derive an input for the cognitive engine, it is not the subject of this work. Instead, we leverage the already large volume of existing research. In this regard, the work in [64] studied modulation classification based on second and higher order time variant periodic cumulant function of the sensed signal for which it is required a prior knowledge of the signal parameters. Authors in [65] used the same framework to perform signal pre-processing, along with utilizing artificial neural networks to address the issues associated with classification when the signal parameters are unknown. The work in [66] proposed a fully automated modulation classification scheme which employs two stages of signal processing to classify the modulation of an incoming signal. In this chapter we assume that each SU applies the technique proposed in [66] on the nearest primary link to infer its used modulation scheme.

Leveraging the use of adaptive modulation in the PN will allow us not only to assess the effects of the SN on the PN, but will also allow us to establish the PN interference threshold. As shown in [67], the use of adaptive modulation allows for the background noise to increase up to a certain level before the average throughput in a network starts to decrease. In the context of underlay DSA where the PN does not exchange any information with the SN, the interference imposed on the PN by an underlay-transmitting CR

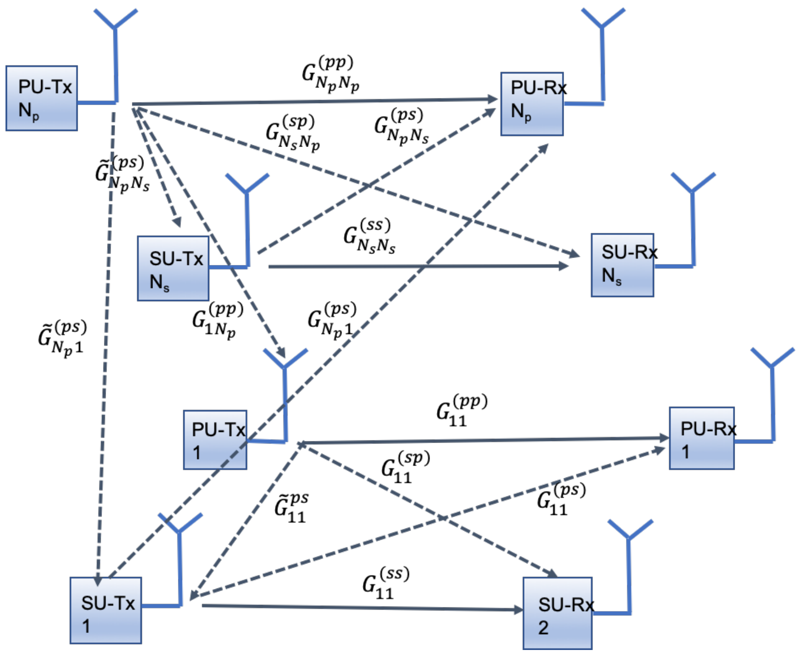

Figure 2.1: Considered network model composed of N_p primary transceivers and N_s cognitive radio users.

can be seen as a background noise for the PN, that can be increased up to a level which does not affect the average throughput in the primary network. Therefore, a CR can become aware about the interference that is creating on the primary link and decide on its transmit power by inferring the experienced throughput of the primary link (which is equivalent to the full AMC mode used) and detecting any change in it.

The main contribution of this chapter resides in presenting an underlay DSA technique to infer, without tapping into feedback or control channels from another network, the modulation order and the channel coding rate (and accordingly the effective bit rate) used in the transmission of the other network (here primary network), and to leverage this inference in the realization of a fully autonomous and distributed underlay DSA scheme. This ability to estimate the effective throughput enables a finer control knob for a more accurate power allocation with less harmful effect on PN transmissions as compared to techniques that rely only on the estimation of modulation order (from applying signal processing on the transmission waveform).

2.2 System Model

We consider a primary network with N_P active primary links coexisting with N_S active secondary links, with both networks transmitting over the same frequency band. The system model is shown in Fig.2.1, where $G_{ij}^{(ps)}$ is the path gain from jth. transmitting SU to the receiver in ith. primary link (denoted as cross-channel gain), $G_{ij}^{(ss)}$ is the path gain from the transmitter in the jth. secondary link to the receiver in the ith. secondary link, $G_{ij}^{(sp)}$ is the path gain from the transmitter in jth. primary link to the receiver in ith. SU link, $G_{ii}^{(pp)}$ is the path gain from the ith. primary transmitter to its corresponding receiver, and $G_{ii}^{(ss)}$ is the path gain from the ith. secondary transmitter to its corresponding receiver. While, $\tilde{G}_{ij}^{(ps)}$ is the path gain from the transmitter in jth. secondary link to the transmitter in ith. primary link.

In this section we focus on describing the operation of the PN, which is incumbent to the considered radio spectrum band. Section 2.4 will present the underlay DSA scheme implemented in the SN. We assume that the PUs receive service from N_{PBS} transmitting base stations (BSs), and we call the ratio N_P/N_{PBS} as the primary network load. Each PU is assigned to the base station that presents the best channel gain. In addition, AMC is used in all transmissions (primary and secondary networks). This means that a transmitter has information about its link quality, in terms of SINR, and based on this assessment chooses, from a set of options, the modulation scheme and channel coding rate that results in highest throughput while at the same time meeting a maximum bit error rate (BER) limit.

Let $P_i^{(p)}$ denotes the transmit power in the ith. active primary link ($i = 1, 2, \ldots, N_P$). Then, in the absence of the SN, the SINR in the ith. primary link (which is used to decide on the AMC mode) can be written as,

$$\gamma_i^{(p)} = \frac{G_{ii}^{(pp)} P_i^{(p)}}{\sum\limits_{j \neq i} G_{ij}^{(pp)} P_j^{(p)} + \sigma_p^2}, \quad i = 1, 2, ..., N_P, \tag{2.1}$$

where σ_p^2 is the background noise power.

In addition to AMC, without loss of generality, we adopt for the primary network the variable transmit power allocation algorithm proposed in [67]. This is an iterative power control algorithm that converges to a global optimum solution that maximizes the product of SINRs across all active links. In the algorithm, the transmit power at the ith. primary link is updated as,

$$P_i^{(p)} \longleftarrow \left(\sum_{j \neq i} \frac{G_{ji}^{(pp)}}{\sum_{m \neq j} G_{jm}^{(pp)} P_m^{(p)} + \sigma_p^2} \right)^{-1}. \tag{2.2}$$

2.2.1 SU Transmission Effect Assessment on the PU: Analytical Framework

In this section, we develop an analytical framework to explore, under a fully autonomous and distributed transmission scenario, the possibility of closed-form analytical estimation of the interference created by a transmitting CR to its nearest PU link. It is assumed that the PUs are oblivious to the existence of the SUs and treat the interference from transmitting CRs as additional noise at their receiver. As mentioned earlier, we consider a practical scenario where there is no communication channel dedicated for the PUs to send any side information (e.g., $G_{ij}^{(ps)}$) to the CRs in order to facilitate their interference control to the PN.

Similar to our proposed technique to be presented inhere, we assume two stage of operations in the SN. During the first stage, every SU in the SN listens to the transmissions in the PN and observes the received signal power. In the second stage, every SU broadcasts a probing signal with the same power to interfere with transmissions in the PN. Since PUs deploy transmit power and rate adaptation upon receiving an interference signal, every PU's receiver sends back a control signal to its corresponding transmitter to adapt its transmit power and rate (in specific, AMC mode in this work) accordingly. Finally, the PUs transmit adaptations are observed by the SUs. We assume that SUs know the primary transmission protocol and are able to synchronize their operation with the primary transmissions, and that all the channel gains involved in Fig. 2.1 remain constant during this process.

Let $P_i^{(p)}[0]$ denotes the initial transmit power in the ith. active primary link ($i = 1, 2, \ldots, N_P$) while SUs listen to the transmissions in the PN. The received signal power at this stage (i.e. listening stage) at ith. SU transmitter can be written as $S_i[0] = \sum_{j=1}^{N_P} \tilde{G}_{ij}^{(ps)} P_j^{(p)}[0]$. This equation further can be separated into two parts: one for the power received from the nearest PU transmitter and a second component for the received power from the rest of active PUs in the system, as in,

$$S_i[0] = \tilde{G}_{in}^{(ps)} P_n^{(p)}[0] + \sum_{\substack{j=1 \\ j \neq n}}^{N_P} \tilde{G}_{ij}^{(ps)} P_j^{(p)}[0], \tag{2.3}$$

where $\tilde{G}_{in}^{(ps)} P_n^{(p)}[0]$ shows the power received from the nearest PU active link (the subscript n is used to highlight, among all active transmission links in the PN, the nearest link to the ith. SU). Considering the same probing signal power for all SUs to be as $P^{(s)}$, the observed power at the ith. SU transmitter during the probing stage can be also written as:

$$S_i[1] = \tilde{G}_{in}^{(ps)} P_n^{(p)}[1] + \sum_{\substack{j=1 \\ j \neq n}}^{N_P} \tilde{G}_{ij}^{(ps)} P_j^{(p)}[1] + I_{ss}, \tag{2.4}$$

where $I_{ss} = \sum_{j=1, j \neq i}^{N_S} G_{ij}^{(ss)} P^{(s)}$ denotes the total interference from the SN to the ith. transmitting SU, and $P_j^{(p)}[1]$ denotes, for the jth. PU transmit link, the new adapted transmit power. Without loss of generality it can be assumed that $P_i^{(p)}[0] > 0$, and thus $S_i[0] > 0$, because if $P_i^{(p)}[0] = 0$, the SU can simply transmits as the spectrum band is unoccupied; and the estimation of the created interference becomes unnecessary. The received signal powers during the listening and probing stage contain the information about the cross-channel gain between ith. SU transmitter and its corresponding nearest PU receiver $G_{in}^{(ps)}$. As mentioned earlier, the cross-channel gain can be used at the SU to assess its effect of transmission on the PU. Thus, in the following we explore whether each SU can determine $G_{in}^{(ps)}$ using its observations. By dividing the signal powers received at the ith. SU transmitter across two listening and probing stages $S_i[0]$ and $S_i[1]$, we have:

$$\frac{S_i[1]}{S_i[0]} = \frac{\tilde{G}_{in}^{(ps)} P_n^{(p)}[1] + \sum_{j \neq n}^{N_P} \tilde{G}_{ij}^{(ps)} P_j^{(p)}[1] + I_{ss}}{\tilde{G}_{in}^{(ps)} P_n^{(p)}[0] + \sum_{j=1, j \neq n}^{N_P} \tilde{G}_{ij}^{(ps)} P_j^{(p)}[0]} \tag{2.5}$$

On the other hand, based on the assumption we made in this chapter, each SU is able to perfectly estimate the adapted modulation order (number of bits per modulation symbols) of its nearest PU link. Due to the use of AMC at every transmission link in the system, the relation between modulation order and the experienced SINR at ith. primary link receiver can be expressed as [67**?**],

$$M_i^{(p)} = \log_2(1 + k\gamma_i^{(p)}), \tag{2.6}$$

where $M_i^{(p)}$ is the ith. PU's modulation order, k is the inverse SNR gap constant $k = \frac{1.5}{-\ln(5BER)}$ which depends on a target maximum transmit bit error rate (BER) requirement, and $\gamma_i^{(p)}$ is the SINR at the receiver of this link. During the listening stage, the $\gamma_i^{(p)}$ can be expressed as,

$$\gamma_i^{(p)}[0] = \frac{G_{ii}^{(pp)} P_i^{(p)}[0]}{I_{pp}[0] + \sigma_p^2}, \tag{2.7}$$

where $I_{pp}[0] = \sum_{j=1, j \neq i}^{N_P} G_{ij}^{(pp)} P_j^{(p)}[0]$ denotes the interference from the PN to the ith. primary link, and σ_p^2 is the background noise power. Using (2.6) and (2.7), $P_i^{(p)}[0]$ can be expressed as follows,

$$P_i^{(p)}[0] = \frac{(2^{M_i^{(p)}[0]} - 1)(I_{pp}[0] + \sigma_p^2)}{kG_{ii}^{(pp)}}, \tag{2.8}$$

where $M_i^{(p)}[0]$ denotes the PU's modulation order. During the probing stage, due to the additional interference created by the SN, $\gamma_i^{(p)}[0]$ is changed

$$\frac{S_i[1]}{S_i[0]} = \frac{\tilde{G}_{in}^{(ps)} \frac{(2^{M_i^{(p)}[1]}-1)(I_{pp}[1]+I_{ps}+\sigma_p^2)}{kG_{ii}^{(pp)}} + \sum_{\substack{j=1 \\ j\neq n}}^{N_P} \tilde{G}_{ij}^{(ps)} P_j^{(p)}[1] + I_{ss}}{\tilde{G}_{in}^{(ps)} \frac{(2^{M_i^{(p)}[0]}-1))(I_p[0]+\sigma_p^2)}{kG_{ii}^{(pp)}} + \sum_{\substack{j=1 \\ j\neq n}}^{N_P} \tilde{G}_{ij}^{(ps)} P_j^{(p)}[0]}, \tag{2.11}$$

to be:

$$\gamma_i^{(p)}[1] = \frac{G_{ii}^{(pp)} P_i^{(p)}[1]}{I_{pp}[1] + \sum_{j=1}^{N_S} G_{ij}^{(ps)} P^{(s)} + \sigma_p^2}, \tag{2.9}$$

where $I_{pp}[1] = \sum_{j=1, j\neq i}^{N_P} G_{ij}^{(pp)} P_j^{(p)}[1]$ denotes the interference from the PN to the ith. primary link. Using (2.6) and (2.9), $P_i^{(p)}[1]$ can be expressed as:

$$P_i^{(p)}[1] = \frac{(2^{M_i^{(p)}[1]} - 1)\left(I_{pp}[1] + \sum_{j=1}^{N_S} G_{ij}^{(ps)} P^{(s)} + \sigma_p^2 \right)}{kG_{ii}^{(pp)}}, \tag{2.10}$$

where $M_i^{(p)}[1]$ denotes the PU's new adapted modulation order at the probing stage. By substituting $P_n^{(p)}[0]$ with (2.8), and $P_n^{(p)}[1]$ with (2.10), equation (2.5) can be rewritten as (2.12), where $I_{ps} = G_{in}^{(ps)} P^{(s)} + \sum_{j=1, j\neq n}^{N_S} G_{ij}^{(ps)} P^{(s)}$ displays the interference from the SN to the ith. SU's nearest primary link after probing (recall that cross-channel gain is denoted by $G_{in}^{(ps)}$).

$$\frac{S_i[1]}{S_i[0]} = \frac{\tilde{G}_{in}^{(ps)} \frac{(2^{M(\gamma_i^p[1])}-1)I_{pp}[1]+P^{(s)}G_{in}^{(ps)}+I_{ps}+\sigma_p^2}{kG_{ii}^{(pp)}} + \sum_{j=1, j\neq n}^{N_P} \tilde{G}_{ij}^{(ps)} P_j^{(p)}[1] + I_{ss}}{\tilde{G}_{in}^{(ps)} \frac{(2^{M(\gamma_i^p[0])}-1))I_p[0]+\sigma_p^2}{kG_{ii}^{(pp)}} + \sum_{j=1, j\neq n}^{N_P} \tilde{G}_{ij}^{(ps)} P_j^{(p)}[0]}, \tag{2.12}$$

where $I_{pp}[1] = \sum_{j=1, j\neq n}^{N_P} G_{ij}^{(pp)} P_j^{(p)}[1]$ denotes the interference from the PN to the ith transmitting SU's nearest primary link after probing, $I_{pp}[0] = \sum_{j=1,j\neq n}^{N_P} G_{ij}^{(pp)} P_j^{(p)}[0]$ denotes the interference from the PN to the ith transmitting SU's nearest primary link before probing, $I_{ss} = \sum_{j=1,j\neq i}^{N_S} G_{ij}^{(ss)} P^{(s)}$ denotes the interference from the SN to the ith transmitting SU, and $I_{ps} = \sum_{j=1,j\neq n}^{N_S} G_{ij}^{(ps)} P^{(s)}$ displays the interference from the SN to the ith transmitting SU's nearest primary link after probing.

In the particular case when the PN and the SN have each only one link, (2.12) can be simplified as:

$$\frac{S_i[1]}{S_i[0]} = \frac{(2^{M_i^{(p)}[1]} - 1)(G_{in}^{(ps)} P^{(s)} + \sigma_p^2)}{(2^{M_i^{(p)}[0]} - 1)\sigma_p^2}, \tag{2.13}$$

Provided that the σ_p^2 is known by the SU, the ratio $\frac{S_i[1]}{S_i[0]}$ can be used to estimate the cross-channel gain and subsequently the interference from the SU to the PU (Note that $M_i^{(p)}[1]$ and $M_i^{(p)}[0]$ are known at the SU transmitter). Yet, in the general case of having multiple transmission links in both networks, (2.12) is the appropriate expression which requires from the SUs knowledge of too many system variables (the gains, etc.). Under the assumption of a fully autonomous network, it is almost impossible to achieve all involved parameters and obtain a closed solution to this problem (and in specific estimate for the SUs their corresponding cross-channel gains). Therefore we resort to the use of artificial neural network to solve the problem of SUs assessing their effect on the PN. The rational for this decision rests on the artificial neural network's property of being known as universal function approximator and their ability to implicitly extract, during the learning (training) process, the interrelation between the system variables. Leveraging these characteristics of artificial neural networks in our proposed technique allows for the SUs to assess their transmission effect on the PN without the need to calculate intermediate magnitudes (e.g. the cross-channel gain(s)).

2.3 Leveraging Adaptive Modulation and Coding in Underlay DSA

Before presenting our proposed underlay DSA technique, in this Section we outline the main ideas on how AMC can be leveraged to address the two main challenges associated with underlay DSA: How to establish the interference threshold in the PN and how SUs can autonomously become aware of the interference they create on the PN. As previously noted, AMC (or, as also called, link adaptation) has been used during the past two decades in practically all high-performance wireless communications standards and, as such, is assumed to be used in both the primary and secondary networks in this work. In this section we summarize some main features of AMC that are relevant to our fully autonomous and distributed underlay DSA scheme.

Fig. 2.2, obtained using the Matlab LTE Link Level Simulator from TU-Wien [68], shows the throughput versus signal-to-noise ratio (SNR) performance for the LTE system that will serve without loss of generality as the assumed AMC setup for the rest of this work (setup details for the simulation results shown in Fig. 2.2 are discussed in Section **??**). In LTE, AMC consists of 15 different modes (each for a different "Channel Quality Indicator" - CQI) based on three possible modulation schemes which will be called "*type 0*" for QPSK (used for the smaller SNR regime), "*type 1*" for 16QAM (used at intermediate SNRs), and "*type 2*" for 64QAM (used for the larger SNRs). In AMC, alongside the modulation order, channel coding rate is also adapted [69, 70]. Fig. 2.2 shows the throughput of one LTE resource block achieved for each AMC mode (each curve is labeled with the corresponding CQI value and AMC mode settings, formed by the modulation type and channel coding rate) and the overall performance curve of the AMC scheme, where the modulation type and code rate are chosen to maximize throughput but with a constraint on the block error rate (BLER) not to exceed 10%. During transmission, the transmitter chooses the AMC mode with maximum throughput at the estimated SNR of the link.

As was discussed in [67], the use of AMC in conjunction with transmit power control allows the background noise to increase up to a maximum value without significantly affecting the network average throughput. In

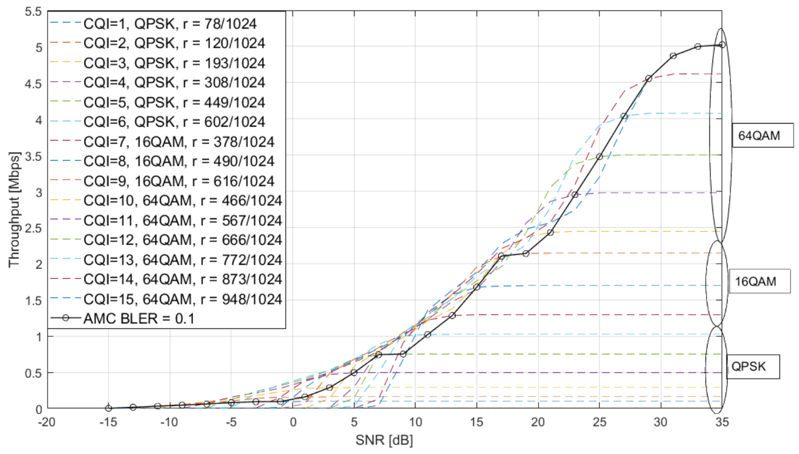

FIGURE 2.2: Throughput of one resource block as a function of SNR for the AMC scheme of an LTE system .

the context of underlay DSA, this maximum noise value can be interpreted as the maximum value for the combined powers of background noise and interference from the SN and the rest of the PN. To see this important point in detail, consider Fig. 2.3, which is an expanded version of Fig. 5 in [67], now for different network loads (N_P/N_{PBS}) when using the LTE AMC setup just described and the power control algorithm from [67]. The figure shows the average throughput achieved in the PN by itself (the presence of an SN is not included in this result) as the background noise power increases. On the top, the figure shows the property associated with the use of adaptive modulation that for all network loads the average throughput remains approximately constant until noise power becomes sufficiently significant. This is not a trivial observation as networks with a higher load are operating in a regime more influenced by the interference rather than the noise, but it is clear from the results that adaptive modulation manages to maintain a balance between interference and noise-dominated operation. The bottom of Fig. 2.3 shows as a function of noise power, the change in throughput relative to the throughput at the lowest noise power. The result exposes the remarkable property that the interference that would be imposed by the SN, which can be considered by a PN that is unaware of the presence of another network as part of the background noise, will not significantly affect the average throughput in the primary network as long as the combined SN interference, the interference by other primary links and actual background noise remains below a threshold approximately equal to -85 dBm (although this number somewhat depends on the network load).

Moreover, relative throughput change starts to decrease at approximately the same value of noise power for all network loads (around -90 dBm). This is a consequence of the link adaptation performed through AMC. Moreover, throughput relatively decreases faster with smaller network loads. We believe that this is because at smaller network loads, interference across the network is lower and a larger ratio of transmissions use the less resilient higher rate modulation types.

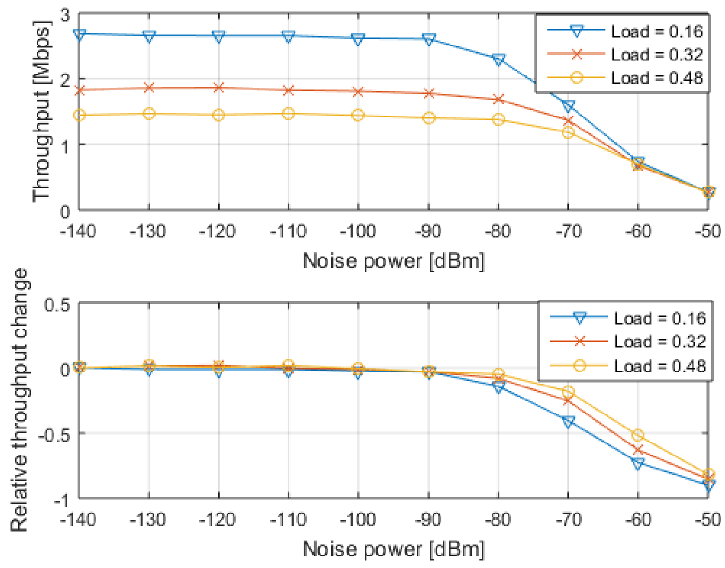

Figure 2.3: Throughput vs. noise power in the primary network.

While AMC entails the adaptation of both modulation order and channel coding rate, it can be seen in Fig. 2.2 that the modulation order provides a coarse adaptation and that the channel coding rate enables a finer adaptation within each of the choices for modulation order. Moreover, an important difference between modulation order and channel coding rate adaptations is that while it is possible for a passive "listener" of the AMC transmission to infer the modulation order through the use of modulation classification signal processing, it is not possible to infer the channel coding rate. Indeed, there exists a large body of research in the area of modulation classification with some representative works briefly discussed in Sect. 1 (e.g. [64–66]). Consequently, the techniques that existed before our work have been limited to use only the coarse information derived from the modulation order (e.g. [59, 60]) or to rely on the sharing of information between the PN and the SN through the SUs accessing the control feedback channel in the PN to learn the finer information associated with the channel coding rate used in a primary link (e.g. [54]). As will be seen, our proposed technique is able to overcome the limitation of inferring the channel coding rate without exchange of information between the PN and the SN and, as such, be able to use the fine-grained information provided by channel coding rate without the SN tapping into any control channel of the PN.

We finally highlight that, as can be seen in Fig. 2.2, when the noise power increases (or equivalently the SN interference increases) the effect of AMC operation will be to change the AMC mode to one associated with a smaller CQI. At the same time, the modulation scheme with the smallest order is used for the smallest operating SINRs (because the transmission of less bits per symbol is more resilient to interference and noise). As the interference from secondary transmissions increases, primary links that are already using this modulation scheme in the absence of secondary transmissions will not switch to other modulation schemes because there is no other modulation scheme with fewer bits per symbols (with smallest associated CQI) to switch to. This means that transmissions from an SU that otherwise would generate a change in modulation scheme would not result in any change when the nearest primary link is already transmitting with the modulation scheme with smallest bits per symbol.

2.4 Autonomous and Distributed Underlay DSA for Cognitive Radio Networks

We now present the main contribution of this chapter: a fully autonomous and distributed underlay DSA technique for a secondary CR network. We assume that the transmitting SUs are randomly located within the area covered by the primary network and that each of them has assigned a receiving SU randomly located within a circular region around the transmitter. The operation of the SN is fully autonomous and ad-hoc. This means that SUs do not rely on any exchange of information with the PN and with other SUs (other than between transmitter-receiver pairs) and that the transmission control algorithm in the SN needs to be distributed. While there is no information exchange between primary and secondary networks, it is assumed that the SN has knowledge of the underlying timing operation in the PN so as to allow CRs to sense at appropriate times.

Fully autonomous operation implies that the primary and secondary networks operate as being unaware of the other (except for the above mild timing assumption), considering the other network transmissions as out-of-network interference akin to background noise. Then, when adding an underlay SN, the SINR at the receiver of the ith. PN link now becomes,

$$\gamma_i^p = \frac{G_{ii}^{(pp)} P_i^{(p)}}{\sum_{j \neq i}^{N_P} G_{ij}^{(pp)} P_j^{(p)} + \sum_{j=1}^{N_S} G_{ij}^{(ps)} P_j^{(s)} + \sigma^2}, \tag{2.14}$$

where $P_j^{(s)}$ is the transmit power from the jth. transmitting SU and $G_{ij}^{(ps)}$ is the path gain from a transmitting SU j to a PU i. Likewise, it is assumed that transmissions on the SN also make use of AMC, which is configured based on the corresponding link SINR. For this, the SINR, $\gamma_i^{(s)}$, at the receiver of the ith. SN link is

$$\gamma_i^{(s)} = \frac{G_{ii}^{(ss)} P_i^{(s)}}{\sum_{j \neq i}^{N_S} G_{ij}^{(ss)} P_j^{(s)} + \sum_{j=1}^{N_P} G_{ij}^{(sp)} P_j^{(p)} + \sigma^2}, \tag{2.15}$$

where $G_{ij}^{(ss)}$ is the path gain from the transmitter in the j-th secondary link to the receiver in the i-th secondary link, and $G_{ij}^{(sp)}$ is the path gain from j-th BS to i-th SU. All transmissions are assumed to be over a flat quasi-static fading channel that is considered constant during a sensing and transmission period. This setup assumed for the SN is general, yet practical, as it is applicable to numerous wireless cognitive ad-hoc networks scenarios.

In the proposed underlay DSA technique, a cognitive engine at each SN transmitter learns the functional model of the interaction between the secondary and primary networks. Often, analytical models have been used to characterize the performance of the SN. For example, in [71] the BER performance of different modulation orders have been characterized using analytical models. However, we think that deriving an analytical model to devise the throughput and specifically channel coding rate in primary link is very challenging issue, specially in nonideal scenario of real wireless network deployments, and involves making several assumptions such as presence of ideal channel gains between nodes in the system. Black-box modeling is an alternative approach to analytical models which consider analyzing the relation between input and output of a system and aims at building a predictor to estimate output values for unforeseen inputs and variations of the system configurations. In this work we will follow a black-box modeling approach.

Neural Networks (NNs) applied to black-box modeling have become increasingly popular as general purpose function approximators and, specifically, for dynamic system modeling [72]. Neural networks have been successfully applied to a number of modeling and time series prediction tasks. Due to the inherent capability of neural networks in modeling nonlinear systems and their higher robustness to noise, they frequently outperform standard linear techniques when the time series is noisy and the dynamical system that generated the time series is nonlinear [73]. There is a growing number of works that have applied neural networks for various communication tasks such as channel decoding, estimating the features of the user channels and predicting the anomalies for wireless sensor networks [74–76]. For CRs, the feed forward neural network has been used in predicting the spectrum occupancy status [77] and designing a medium access control (MAC) protocol [78].

Due to the adaptation of SUs transmission to the PN interference threshold, an underlay network can be seen as an example of a dynamic system, and the throughput in the PU can be also seen as a time series with a temporal dependency. As a result, we have considered a neural network-based cognitive engine to in specific predict as a time series data the throughput in PU and characterize the behavior of such dynamic system. In the case of one-step-ahead time series prediction tasks, since only the estimation of the next sample value of a time series is required, without feeding back the output as a new input to the model, the input contains only actual sample points of the time series. While considering multi-step-ahead or long-term prediction, the neural network model's output should be fed back to the model as a new input for a finite number of time steps [79]. In this case, the components of this input to the model, previously composed of actual sample points of the time series, are gradually replaced by previously predicted values. As a result, the multi-step-ahead prediction task is converted to a dynamic modeling task. In this case, the neural network model behaves as an autonomous system and tries to recursively emulate the dynamic behavior of the system that generated the nonlinear time series [80]. Compared to the one-step-ahead prediction, multi-step-ahead prediction and dynamic modeling are much more complex to deal with. However, neural networks models and in particular recurrent neural architectures play an important role in dealing with these complex tasks [81]. Elman introduced in [82] a class of recurrent neural models called simple recurrent networks (SRNs) which are essentially feedforward in the signal-flow structure with a few local and/or global feedback loops. A time delay neural network (TDNN), which is an adapted version of a feedforward multilayer perceptron (MLP)-like networks with an input tapped-delay line, can be used to process time series [81]. In the case of long-term predictions, a feedforward TDNN model will eventually behave similarly to the SRN architecture, since a global loop is needed to feed back the current estimated value into the model's input. Temporal gradient-based variants of the backpropagation algorithm are usually used to train the aforementioned recurrent neural networks [83]. However, training using gradient-based learning algorithms can be quite difficult in the case of systems with a long time temporal dependencies in their input-output signals [84]. In [85], the authors claimed that such training is more effective in a class of simple recurrent network model called Nonlinear Autoregressive with eXogenous input (NARX) [86–88] than in simple MLP-based recurrent models. This class of neural networks were proven to be powerful in pattern recognition and classification applications as well [89, 90]. In this chapter, the architectural approach proposed for modeling underlay system was chosen based upon the NARX neural network structure. Specifically, we choose the NARX neural network to implement the cognition task of cognitive engine in each SU.

With a topology as shown in Fig. 2.4 for the case of one hidden layer network, the NARX neural network output can be mathematically represented

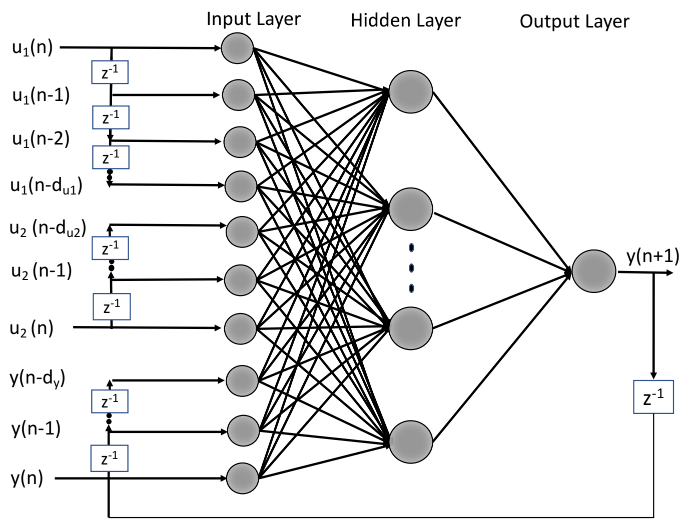

FIGURE 2.4: NARX neural network with two delayed inputs and one delayed output.

as [91],

$$\begin{aligned} y(n+1) = f\Big(&y(n), y(n-1), \ldots, y(n-d_y); \\ &u_1(n), u_1(n-1), \ldots, u_1(n-d_{u_1}); \\ &u_2(n), u_2(n-1), \ldots, u_2(n-d_{u_2})\Big), \end{aligned} \tag{2.16}$$

where $u(n)$ and $y(n)$ denote, respectively, the input and output of the model at discrete time step n, and $d_{u_1} \geq 1$, $d_{u_2} \geq 1$ and $d_y \geq 1$, $d_{u_1} \geq d_y$, $d_{u_2} \geq d_y$ are the input and output discrete delays, respectively. The nonlinear mapping $f(\cdot)$ in (2.16) can be approximated, for example, by a standard feedforward multilayer neural network. If the non-linear mapping can be learned accurately by a neural network of moderate size (measured in terms of number of layers and number of artificial neurons in each layer), the resource allocation based on the output of the NARX neural network can be done in real time, since passing the input through the neural network only requires a small number of simple operations.

In Fig. 2.4 each circle represents an "artificial neuron", an elementary operation unit in the NARX neural network model which performs an operation

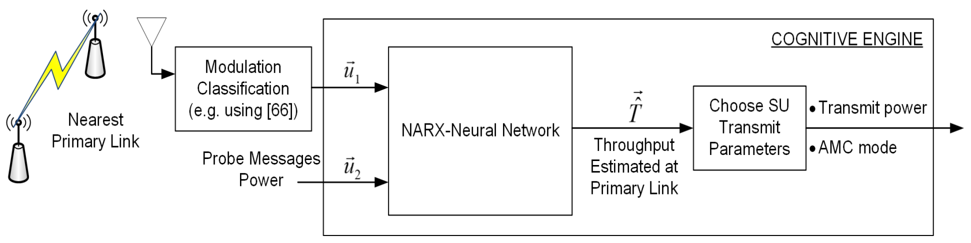

FIGURE 2.5: Block diagram of an SU with a cognitive engine based on the NARX neural network.

on its inputs given by,

$$z = \phi\Big(\sum_i w_i x_i + b\Big), \tag{2.17}$$

where z is the output of the neuron, x_i is the ith. input, which is multiplied by the weighting factor w_i, b is a bias applied to the neuron (not explicitly shown in Fig. 2.4) and $\phi(\cdot)$ is called the "activation function". In our implementation of the NARX neural netowrk, this activation function $\phi(\cdot)$ for the hidden layer is a sigmoid function while the activation function used for the output layer is linear (the input layer is not truly formed by artificial neurons but rather it is conventionally included as a representation of the connections of inputs into the neural network).

Fig. 2.5 illustrates the block diagram of an SU with its cognitive engine based on the NARX neural network, as well as other processing steps associated with the inputs and output of the neural network. The function of this cognitive engine will be to aid in the setting of power control and AMC parameters for a transmitting SU, by predicting for different transmit settings the throughput $\hat{T}$ (equivalently the CQI or AMC mode) of the nearest PN link. Under our imposed practical condition of no exchange of information between the primary and secondary networks, a CR can only estimate the modulation order, after performing modulation classification signal processing on the PN transmissions, and immediately has no direct way to know the coding rate in use (this is, unless accessing the PN's feedback channel, which would violate our condition). Moreover, practical limitations further dictate that the modulation classification can only be performed on the one primary transmission that it is being received with strongest power (usually this is the closest one), making the other transmissions be interference. It is assumed that the estimation of modulation type does not rely on the SU accessing any information from the PN feedback channel and is error free using any of the methods existing in the literature (e.g. [65] or [66]). Since the modulation order setting of a primary link provides a coarse indication of its SINR, this is, of the effect that an SU transmission at some power level has on the primary link, we configure the NARX neural network cognitive engine to have as inputs the most recent values of assigned transmit power levels at the SU ($u_1(n)$) and, corresponding to these transmit power levels, the modulation order ($u_2(n)$) being used in the closest primary link that is inferred using a modulation classification technique. The output of the NARX neural network cognitive engine is the predicted throughput $\hat{T}(n)$ at the primary link closest to the transmitting CR. The predicted throughput on a primary link corresponds to a choice of AMC mode.

The proposed NARX neural network undergoes a training process to determine the values of its weights and biases. In this training process, the weight and bias values are updated according to the Levenberg-Marquardt optimization method. This method involves a back-propagation algorithm to compute the gradients of the prediction error corresponding to the artificial neurons [92]. It is known that in the case of function approximation problems, for the neural networks containing up to a few hundred weights, the Levenberg-Marquardt algorithm will have the fastest convergence [93]. Mean square error (MSE) was chosen as the prediction error metric:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(e_i)^2 = \frac{1}{N}\sum_{i=1}^{N}(T_i - \hat{T}_i)^2, \tag{2.18}$$

where N is the number of neurons in the hidden layer, T_i and $\hat{T}_i$ are target and predicted values, respectively. In order to find the number of hidden nodes and depth of the tap-delay lines in the NARX neural network, we perform a through set of experiments with different configurations. The best performing structure was found to be with 50 hidden nodes and 7 time delay steps. Fig. 2.6 illustrates the performance of the NARX neural network for a primary network load equal to 0.64. Further details regarding the generation of the target prediction values T_i used during training are provided in the next Section.

For the complete system operation, first the SUs avoid transmission while

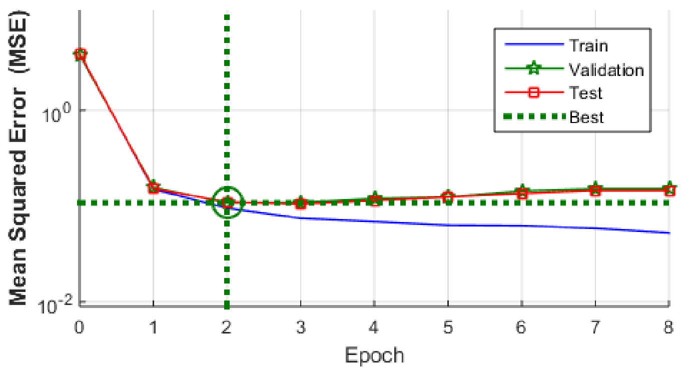

Figure 2.6: Throughput prediction performance of the NARX neural network. Best validation performance is 0.10825 at epoch 2.

the PN initially adjusts its power and AMC parameters using the iterative power control algorithm (2.2). During this stage, the SUs listen to the PN transmission and infer the modulation order used by their closest primary link. At this stage, the SUs obtain an estimate of the modulation order used by their corresponding nearest primary link when the SN is not transmitting. After this, each SU proceeds to send a series of short probe messages configured with different transmit powers. Each SU performs modulation classification for each probe message on its nearest primary link. The sequence of probe message transmit powers and ensuing modulation orders are fed to the NARX neural network, which provides the sequence of corresponding estimated throughput values at the nearest primary link, $\hat{T}$. Note that this sequence of estimated throughput values includes the one with no transmissions from the SN. Since estimating the throughput is equivalent to estimating the CQI and the corresponding AMC mode, the NARX neural network is able to provide an estimate of the nearest primary link SINR with a finer resolution than what could be derived from the modulation classification alone that is present at its input.

Furthermore, the SU can use the sequence of estimated throughput values to infer what would be the effect of its transmission on the nearest primary link by comparing the change in throughput value against that without the SN transmission. As seen in Fig. 2.5, the SU uses this information to find its own transmission parameters. We will consider two approaches for the SUs to choose transmit settings. In the first one, the SU chooses the maximum transmit power value that is estimated to not lead to a change in modulation order at its nearest primary link. In the second approach, the SU chooses the maximum transmit power value that keeps the relative change in throughput of its nearest primary link below a predetermined limit. Note that the second approach differs from the first one in that it fully uses the advantage provided by the NARX neural network in providing the finer resolution inference on the nearest primary link full AMC mode, instead of the coarser inference on modulation order that the first scheme is based on. Additionally, for both approaches, those SUs that estimate that their closest primary link is transmitting in the lowest rate AMC mode when the SN is not transmitting (because of already experiencing a very low SINR, likely leaving no room for added interference from the SN) are prevented from transmitting. This guarantees that the reduction in the average rate of the primary link experiencing the poorest channel quality (CQI equal to 1) is minimized.

Going back to Fig. 2.2 helps to gain an intuition into the NARX neural network cognitive engine operation. The NARX neural network receives as one input the possible SU's transmit power levels organized in sequence and, as another input, the corresponding modulation order sensed from the nearest primary link. During training, the NARX neural network learns to predict the throughput values at the nearest primary link that correspond to the two input sequences. Considering (2.14), we can think that the sequence of transmit power values that is input to the NARX neural network will yield a range of interference values, which will correspond to a "segment" of SINR values in the abscissa of Fig. 2.2. The position of this segment within the range of SINR values depends on the many factors reflected in (2.14) (e.g. primary channel gain, interference from other SUs, etc.) but the NARX neural network has a sense of where the segment is thanks to the reference provided by the sequence of modulation orders at the input (e.g. if for the setup in Fig. 2.2, the sequence of modulation schemes are QPSK and 16QAM, the SINR segment is around 10 dB). During training, the NARX neural network is presented with multiple different such segments from different wireless environment scenarios, eventually learning the throughput vs. SINR AMC performance curve. During operation of the NARX neural network (the testing phase), sensing the sequence of modulation schemes that results from probing SU transmissions with a sequence of possible power settings allows the NARX neural network to localize the segment of SINR values for the nearest primary link (in effect, finding the network scenario presented during training that best matches the existing wireless environment) and, consequently, predicts the corresponding throughput from the AMC performance curve.

2.5 Results

The performance of the presented technique was evaluated through Monte Carlo simulations (150 runs) based on a PN "playground" consisting of a five-by-five BSs grid ($N_{PBS} = 25$) with neighboring base stations separated by a distance of 200 m. To avoid edge effects in the playground, the grid wraps around all its edges. One channel was singled out in the experiment and any of the base stations can have this channel active. In order to reflect realistic AMC settings, we assumed that all transmissions are based on an LTE 2x2 MIMO configuration and that the channel of interest is one resource block with a bandwidth of 180 kHz.

As noted earlier, we assumed that there are N_P active base stations that are using the same channel to communicate with their respectively assigned PU receivers. The location of the N_P PU receivers is determined at random using a uniform distribution with the limitation that no base station could have more than one receiver assigned to it. Also, the receivers were connected to the base station from which they received the strongest signal. Transmit powers in the primary network were limited to the range between -20 and 40 dBm. The transmit power assignment for the ith. active primary link ($i = 1, 2, \ldots, N_P$) follows the same algorithm as in (2.2). Each primary transmission considers the other network transmissions as out of network interference akin to background noise.

In the simulation, the SN consisted of $N_S = 4$ transmit-receive pairs of CRs, with the transmitters placed at random (also with a uniform distribution) on the PN playground, around their respectively assigned transmitter within a distance not exceeding 50 m. In the simulation setup we intended to reflect a situation where the PN had somewhat more capabilities (achieving larger throughput and communication range) than the SN because of being the incumbent to the spectrum band under consideration. Therefore, we assumed that the SUs were smaller devices that communicated with a single omni-directional antenna with transmit power in the range of -30 to 20 dBm. Twenty equally spaced power levels in this range are considered as the set of allowed settings for transmission. The operation of the SN, as mentioned before, is fully autonomous and ad-hoc and the transmission control algorithm is distributed. It is also assumed that the SN has knowledge of the underlying timing operation in the primary network so as to allow CRs to sense and transmit at appropriate times.

All links assumed a path loss model given by $L = 128.1 + 37.6 \log d + 10 + S$ (in dBs), where d is the distance between transmitter and receiver in km, S is the shadowing loss (modeled as a zero-mean Gaussian random variable with 6 dB standard deviation) and the penetration loss is fixed at 10 dB, [94]. Noise power level was set at -130 dBm. It should be noted that all transmissions adopt AMC to adapt their transmission to the quality of their respective link.

Each neural network was trained/validated with a data set of 10000 samples collected from a network simulation with the setup just described. A subset of the data set (7000 samples or 70% of the data set) was used to train the neural network and to present the CRs with new environmental conditions, the rest of data was used to compare the prediction performed by the trained neural network with the actual expected performance in order to encounter new environment conditions. In order to generate training data for the neural network cognitive engine, a comparable SN was devised that maintained the ability to use the estimated modulation order of the nearest primary link but without the cognitive engine shown in Fig. 2.5. Instead, this comparable system implemented a distributed power control algorithm modified to incorporate the modulation order of the nearest primary link. A number of algorithms had been proposed for distributed power control in ad-hoc wireless network. One of the first ones, and the precursor to many related variants, is the Foschini-Miljanic algorithm, [95], which implements an iterative distributed power control process so as to meet a target SINR. We adopted this iterative power allocation algorithm for the alternative SN that generated the data set to train the NARX-NN cognitive engine at each SU. Specifically, power is calculated for a secondary link i at each iteration

m using the update formula,

$$P_i^s[m+1] = \left(\frac{\beta_i}{\gamma_i^s[m]}\right) P_i^s[m], \tag{2.19}$$

where P_i^s is the transmit power, β_i is the target SINR and $\gamma_i^s[m]$ is the actual SINR measured in the mth. iteration which can be calculated through (2.15). The fact that this algorithm associates power control with a target SINR is a useful feature in our case because the target SINR, when met, also determines the modulation order to be used as follows[67],

$$T_i^{(s)} \;=\; \log_2(1 + k\,\gamma_i^{(s)}), \quad i = 1, 2, ..., N_S. \tag{2.20}$$

where $T_i^{(s)}$ and $M(\beta_i) = (1 + k\gamma_i)$ are the ith SU's transmit rate and the number of bits per modulation symbol, respectively. The inverse SNR gap constant $k = \frac{1.5}{-\ln(5BER)}$ depends on a target maximum transmit bit error rate (BER) requirement. Consequently, we can think that instead of having a set of possible [modulation, channel code] pairs, now we have a set of target SINRs to choose from. Let $\mathfrak{B} = \{b_1, b_2, \ldots, b_K\}$ be this set, where target SINRs b_i's are assumed to be sorted in ascending order. Of course, reducing the target SINR will result in decreasing the transmit power. As a consequence, the algorithm provides the mechanisms to both adapt transmit power and AMC settings in a distributed way. Moreover, for fair comparison to the system with the proposed NARX neural network cognitive engine, the transmit power from SUs also needs to be constrained by the goal to not degrade the SINR of the closest primary network link to the extent of reducing the modulation order (not having the NARX neural network, this SN we are comparing against cannot operate based on the use of throughput inferred for the nearest primary link and can only make use of the modulation order estimated from the modulation classification process). Modifying the Foschini-Miljanic algorithm by reducing the target SINR allows to manage this constraint by resulting in a reduction in the SU transmit power. As such, we adopted for the control of CR transmissions in the alternative SN this modified version of the Foschini-Miljanic algorithm, where the SU target SINR is progressively reduced until there is no change in the modulation order of the nearest primary link. We note here that while it is certainly possible to use one of the many existing enhancements to the Foschini-Miljanic algorithm, we chose to use the original version without improvements because its provides a baseline performance measure and because this power control algorithm or a variation of it are not the contribution of our work. Of course, the PUs in this system maintained the power allocation algorithm proposed in [67], as explained in Sect. 2.2. We also note that this benchmark SN constitutes an implementation of the central principles of [59] while also managing practical considerations not addressed therein (e.g. multiple links in the SN and PN, distributed, ad-hoc operation of the SN, etc.), and, as such, serves the purpose of providing an indication of the performance improvements of our proposed technique versus prior works.

Moreover, as briefly mentioned earlier, note that in AMC the modulation order transmitting the smallest number of bits per symbol is used for the smallest operating SINRs (because it is more resilient to interference and noise). When the interference from the SN increases, the primary links that were already using the smallest possible modulation orders when there were no secondary transmissions will not switch to other modulation orders because there is simply no other modulation order with fewer bits per symbols to switch to. This means that transmissions from an SU that otherwise would generate a change in modulation order would not result in any change when the nearest primary link is already transmitting with the modulation order with smallest bits per symbol. Moreover, primary links using this modulation order, do so because their SINR is at the lower range of the operating SINRs, which implies that they are at a link state that likely may not leave much room for added interference from SUs. Because of these reasons, and in the interest of prioritizing the protection of primary links against excessive SN interference, we configured the alternative SN so that a CR will not transmit if it senses that its nearest primary link is using the lowest modulation order when the SN is not transmitting (as explained in Sect. 2.4, our proposed technique implements a mechanisms with the same spirit but based on checking for the smallest CQI at the nearest primary link when the SN is not transmitting, instead of smallest modulation order).

Figs. 2.7 through 2.9 study the throughput performance in the primary and secondary networks for our presented technique and contrast them against other schemes. As indicated in Sect. 2.4, for the presented technique we considered two approaches for the SUs to choose transmit settings. The first approach, labeled in the figures as "*PN+SN- NN Cog. Eng.- Modulation*", features our proposed cognitive engine with the capability for full AMC mode estimation at the nearest primary link, but it makes a limited use of this capability by making the SU choose the maximum transmit power value that is estimated to keep unchanged the modulation order (but not necessarily the channel coding rate) at its nearest primary link. The second approach makes full use of the cognitive engine's throughput (full AMC mode) estimation capabilities at the nearest primary link, by making the SU choose the maximum transmit power value that is estimated to not change the nearest primary link throughput beyond a maximum relative change value. This second approach is itself divided into a case where one probe message is sent for each possible power setting (for a total of twenty probe messages) and a second case that explores a reduction in the overhead from transmitting probe messages which transmits just seven messages and uses interpolation to complete the information for the rest of available transmit power settings. To show how our technique is able to leverage the estimation of the full AMC mode and provide each SU with a fine control over the interference it imposes to the nearest primary transmission link, we obtained results for three different limits on relative average throughput change in the PN: 2%, 5% and 10%. In the figures, we labeled the results when sending all probe messages as "*PN+SN- NN Cog. Eng- 2%- All probe msgs.*" for a limit on relative average throughput change in the PN of 2%, "*PN+SN- NN Cog. Eng- 5%- All probe msgs.*" for a limit on relative average throughput change in the PN of 5%, and "*PN+SN- NN Cog. Eng- 10%- All probe msgs.*" for a limit on relative average throughput change in the PN of 10%. Similarly, for the case when sending seven probe messages, the labels for 2%, 5% and 10% limit on relative average throughput change in the PN are "*PN+SN- NN Cog. Eng- 2%- Seven probe msgs.*", "*PN+SN-NN Cog. Eng- 5%- Seven probe msgs.*" and "*PN+SN- NN Cog. Eng- 10%- Seven probe msgs.*", respectively.

The performance of these realizations of the proposed underlay DSA technique is compared in the figures against other schemes. The first such contrasting scheme, labeled in the figures as "*PN+SN- Adapted Foschini-Miljanic*", is the same system used to collect training data and described earlier in this Section as the alternative SN. Recall that this scheme lacks the NARX neural network cognitive engine's capability to predict the full AMC mode at the nearest primary link. In addition, we considered two schemes that select transmit power for the SUs based on an exhaustive search across all possible setting permutations. In contradiction with our goal to avoid any exchange of information between the primary and secondary networks, these two schemes also incorporate the ability to perfectly know the CQI on the primary links as if the SN had access to the control feedback channels in the PN. The first exhaustive search scheme, labeled "*PN+SN- Exh. search-Max PU throughput*", finds across all possible SUs transmit power permutations, the setting that results in no change in modulation order at any primary link and maximum average throughput in the PN. The second exhaustive search scheme, labeled "*PN+SN- Exh. search-Min PU throughput*", favors the average throughput at the SN by finding across all possible SUs transmit power permutations, the setting that results in no change in modulation order at any primary link and minimum average throughput in the PN. Clearly, the two exhaustive search curves present extreme performance results based on ideal setups. Finally, Fig. 2.7 includes a curve, "*PN without SN*", which shows the average throughput achieved by the PN when the SN is not present.

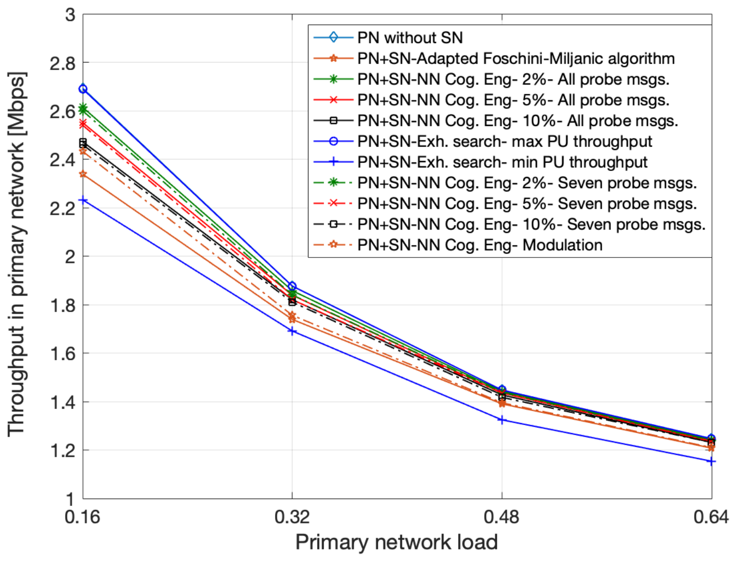

FIGURE 2.7: Average throughput in primary network.

Fig. 2.7 shows the average throughput achieved for the PN as a function of the PN load, N_P/N_{PBS}, while Fig. 2.8 shows the change in this throughput relative to the "PN without SN" case. It can be seen in both figures that the use of the proposed NARX neural network cognitive engine results in SUs transmit settings that reduces the average throughput in the PN much less than the case when the modified Foschini-Miljanic algorithm is used in the SU. Moreover, in the figures, the "PN+SN- Exh. search-Min PU throughput" curve illustrates the extent to which the average throughput in the PN can be affected without changing the modulation order at the nearest primary links (as much as 17%). This indicates that considering only the modulation order provides an initial means for implementing a fully autonomous underlay DSA but with the limitations associated with the coarse indication of the primary links SINR given only by the modulation order. The "PN+SN- Adapted Foschini-Miljanic" and the "PN+SN- NN Cog. Eng.- Modulation" schemes, which both rely on considering modulation order only, show better performance because SU transmit power is chosen in a more conservative way in terms of reducing effects to the primary network, instead of conducting an exhaustive search for the setting that results in minimum average throughput in the PU. Nevertheless, because of relying on modulation order inference only, the "PN+SN- Adapted Foschini-Miljanic" and the "PN+SN- NN Cog. Eng.- Modulation" schemes result in relative reduction in the PN average throughput by as much as 13.5% and 9.5%, respectively. The best performance in terms of controlling the effect of SN transmissions on the PN is achieved with our proposed scheme using the inference of the primary links' full AMC mode (in actuality, the throughput) provided by the NARX neural network cognitive engine. This is seen through the results obtained for the two cases (transmitting either all or seven probe messages) designed on the premise of limiting the maximum relative change in average throughput at the nearest primary link, for which we show results for 2%, 5% and 10% relative change limit. Moreover, these schemes include the means to control as desired the level of SN effect on the PN (by setting the limit maximum relative change). In fact, the "PN+SN- NN Cog. Eng- 2%- All probe messages" along with "PN+SN- NN Cog. Eng- 2%- Seven probe messages" curves exemplify the very fine level of control that is possible to achieve with the proposed approach. Fig. 2.8 shows that this very fine level of control is achieved at all primary network loads, except at the lowest value of 0.16,

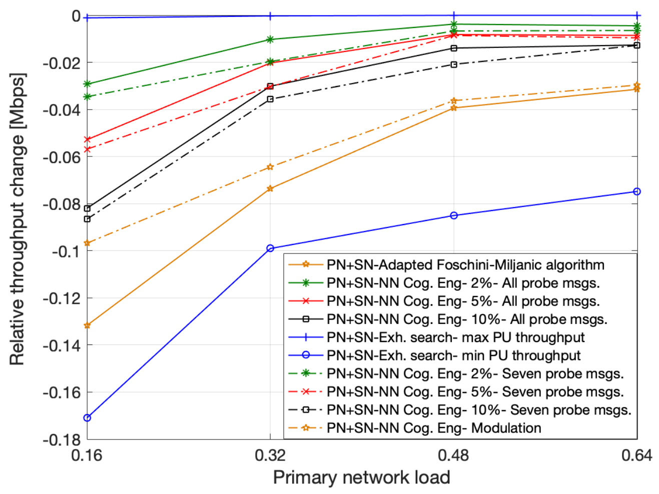

FIGURE 2.8: Relative throughput change in primary network.

when the relative change in average PN throughput exceeds the 2% limit in only 1% when sending all probe messages and in 1.5% when sending seven probe messages. For the rest of cases, only in the case of 5% limit, sending seven probe messages, and at the lowest primary network load, the relative change in PN average throughput is exceeded by just 0.5%. These differences are attributed to errors in estimating the throughput at the PN, which are discussed more in detail later in this Section. Also, the differences are only seen at the smallest PN network load of 0.16 because of the larger sensitivity of the relative change in PN average throughput with lower PN load as was previously highlighted for Fig. 2.3. Fig. 2.8 also shows that the scheme where a fraction of probe messages is used yields significant reduction in the transmission overhead of probe messages (roughly a threefold reduction) without much sacrifice in performance (maximum 3.5% instead of 2% actual achieved relative change in PN throughput compared to 3% maximum change with all the probe messages, and maximum 5.5% instead of 5% actual achieved relative change in PN throughput compared to 5% maximum change with all the probe messages). Finally, the curve "PN+SN- Exh. search-Max PU throughput" coincides with the "PN without SN" curve as the exhaustive search solution that maximizes average PN throughput is essentially the one with no transmissions in the SN (with one caveat to be discussed in Fig. 2.9).

Considering Fig. 2.8 in the context of the bottom plot of Fig. 2.3, we can see that the relative throughput change in the PN when using our proposed technique corresponds to points at equivalent background noise power between -96 and -85 dBm (depending on the target maximum PN relative throughput change setting) just to the left of the "elbow" of the curves in Fig. 2.3. This result confirms that our proposed technique succeeds in its main goal of autonomously and distributively determining the transmit power of the SUs such that the interference they create remains below the maximum threshold value associated with the background noise power levels that a PN with adaptive modulation can sustain. At the same time, the proximity to the "elbow" of the curves in Fig. 2.3 of the equivalent background noise levels generated by the SN indicates that our proposed algorithm is able to find transmit settings for the SUs that will result in as large SN throughput as could be allowed by the PN interference limit.

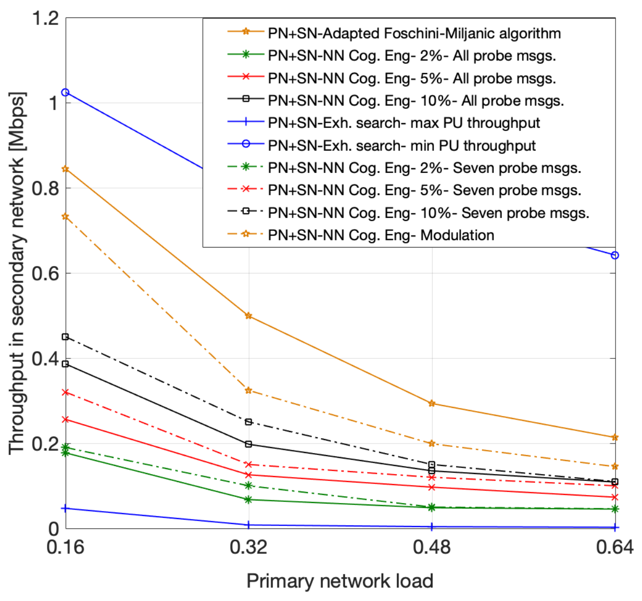

Figure 2.9: Average throughput in secondary network.

Fig. 2.9 shows the average throughput achieved in the SN as a function of the PN load. Naturally, the more a scheme affects the PN throughput, the larger the SN throughput it could achieve. As such, it can be seen that our approach based on a limit maximum PN relative throughout change not only provides the means to control how much the PN is affected by the SN, but also it allows to control how large the average throughput at the SN is desired to be (at the expense of the PN). Even so, the realization with the more restrictive setting for the SN (the one with a maximum PN relative throughput change of 2%) still achieves useful average throughput values

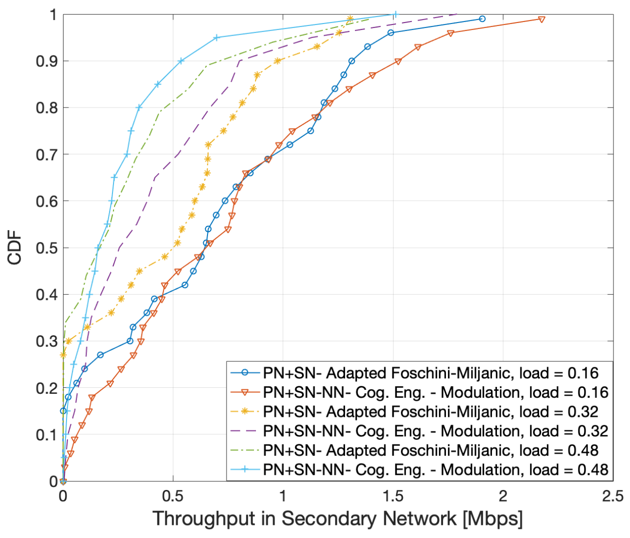

FIGURE 2.10: Cumulative distribution function (CDF) of the throughput in the secondary network.

between 180 and 50 kbps for a channel with 180 kHz bandwidth (in case of transmitting seven probe messages the average throughput at the SN is slightly larger which is consistent with the results in Fig. 2.8). Also, note that at low PN loads, the "PN+SN- Exh. search-Max PU throughput" system shows throughput values that imply transmission in the SN. This does not contradict our earlier statement that this result essentially coincides with the "PN without SN" case. Instead, the transmissions in the SN that are seen in this case correspond to infrequent setups where the SUs are located so far away from the few active primary links (consider that this effect occurs only at very low PN loads) that they can transmit with very low power with no practical effect on the PN.

Fig. 2.10 depicts the cumulative distribution function (CDF) of the throughput in the SN for different primary network loads. This figure presents a perspective that explains an added advantage of the NARX neural network solution compared to the modified Foschini-Miljanic algorithm-based solution. The figure shows that in the case of the SN that uses the modified Foschini-Miljanic algorithm, around 15% of the time SUs will be unable to transmit (throughput is zero) when the PN load equals 0.16 and this number increases to around 30% as the PN load increases. This is because the SN that uses the Foschini-Miljanic algorithm is only able to infer the modulation scheme used in the primary link and not the channel coding rate, which leads to SUs not being able to have a finer assessment of their effect on the PN when the nearest primary link is using a modulation "*type 0*". As a result, and as discussed earlier, in order to protect the PN, those SUs using the modified Foschini-Miljanic scheme for which the nearest primary link use modulation "*type 0*" are blocked from transmitting. In contrast, the proposed technique using NARX neural network, "*PN+SN-NN Cog.Eng.- Modulation*", is able to estimate the finer AMC configuration of coding rate setting, making this protection and the blocking of SU unnecessary (except when the nearest link is using the AMC mode for a lowest rate, which corresponds to CQI=1). Consequently, as seen in Fig. 2.10, the proposed technique increases the transmission opportunities in the SN by the same percentage of time that the SUs are blocked in the case of using the modified Foschini-Miljanic algorithm-based solution.

As just seen, a key advantage of the proposed technique follows from the remarkable ability to estimate the channel coding rate used in a primary link. Therefore, we evaluated the performance of the NARX neural network in estimating the CQI in a primary link (which is equivalent to the full AMC mode consisting of modulation order and channel coding rate). Fig. 2.11 shows as a function of the primary network load the relative frequency of the absolute error when predicting the CQI for the case of transmitting all probe messages. This Figure shows that as the network load increases, the probability of an accurate estimation (prediction absolute error equal to zero) increases and reaches more than 80% for a load equal to 0.48. The Figure shows that, overall, the probability of significant errors when predicting CQI is quite small but, nevertheless, we speculate this to be a factor in the (still small) reduction in PN average throughput and in the small difference at low PN loads between the target maximum relative change in average PN throughput and the actual achieved relative change in PN throughput for our schemes based on target maximum PN relative throughput change.

2.6 Summary

In this chapter we have presented a fully autonomous and distributed underlay DSA technique that is based on a NARX neural network cognitive

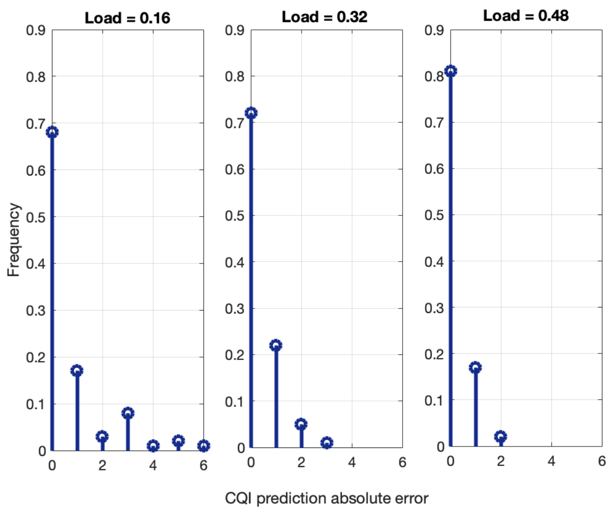

Figure 2.11: CQI estimation performance of the NARX neural network.

engine. The NARX neural network of a transmitting secondary network node learns to use the sensed modulation used in the nearest primary link to predict the effect of its transmission on the nearest primary network link. It does this by predicting the throughput or, equivalently, both the modulation scheme and the channel coding rate resulting from the configuration of adaptive modulation and coding at the nearest primary link. Based on this NARX neural network capability, we presented two variants for the proposed underlay DSA mechanism: one inspired in the current state of the art, where the SUs choose the maximum transmit power value that is estimated to not lead to a change in modulation order at their respective nearest primary link, and a second, more capable mechanism, where the SUs choose the maximum transmit power value that is estimated to not change their respective nearest primary link throughput beyond a chosen maximum relative change value. The performance of the latter proposed underlay DSA mechanism was examined for the cases of sending all or a third of all probe messages.

Simulation results show that the proposed technique is able to accurately predict the modulation scheme and channel coding rate used in a primary link without the need to exchange information between the PN and the SN (e.g. access to feedback channels), while succeeding in its main goal of determining the transmit power of the SUs such that the interference they create remains below the maximum threshold that the primary network can sustain with minimal effect on the average throughput, along with reducing the transmission overhead when sending a fraction of probe messages. At the same time, it was seen that our proposed algorithm is able to find transmit settings for the SUs that will result in as large throughput in the SN as could be allowed by the primary network interference limit. Specifically, for a target PN maximum relative average throughput change of 2% the proposed scheme is able to maintain the PN relative throughput change less than 3% when sending all probe messages, and also less than 3.5% when reducing three times the number of transmitted probe messages, while at the same time achieving useful average throughput values in the secondary network between 180 and 50 kbps for a channel with 180 kHz bandwidth. We also discussed how the ability of our proposed technique to predict the full AMC mode (not just the modulation scheme) results in a significant increase in the transmission opportunities in the SN compared to schemes that only use the modulation classification information.

Chapter 3

Reinforcement Learning-Based Resource Allocation Framework for Heterogeneous Underlay Cognitive Radio Networks

As stated in previous chapters, in underlay DSA both primary and secondary networks simultaneously communicate over the same frequency band. Since in cognitive radio networks the PUs are the incumbent to the spectrum band being used, the SUs must avoid creating more interference than the PU's tolerable threshold. Thus, in these networks transmission power allocation for SUs is the main task needed to be taken into account. The next prioritized task is to maximize the SN performance metric, especially in a heterogeneous scenario where SUs carry different types of traffics, each with differentiated QoS requirements. To represent such a heterogeneous network, we assume that the SU transmission links carry either real-time/streaming video traffic or regular data with markedly different characteristics and requirements. Moreover, QoS is not an appropriate metric to assess the network performance in a heterogeneous network, as it is differentiated across different types of traffic. As a result, we need a uniform metric across different traffic.

In the search for a uniform metric across different traffics, we realized that the quality of experience (QoE) is gaining significant attention as the representative of end-user-centric quality assessment. Consequently, we opted for QoE as the network performance metric and assessing the quality of the delivered traffics in the network. There are a variety of metrics to model subjective QoE. For different types of traffic QoE depends on parameters from different layers of the protocol stack. For this reason, we need to select a metric that models the QoE differently for each type of traffic. Specifically, as we implement an integrated resource allocation, the selected metric not only should incorporate the different parameters of different traffic types, but more importantly should provide a single common measuring scale for dissimilar types of traffic. To address this issue, we choose mean opinion score (MOS) as the metric to model the QoE. The MOS, as the most popular descriptor of human perceived media quality [96]. It has five level quality characterization, as being excellent, good, fair, poor and bad. Importantly, it provides a single common assessment metric across different traffic types and accordingly allows for seamless integrated traffic management and resource allocation across dissimilar types of traffic.

Using this metric, this chapter evaluates the use of reinforcement learning to perform distributed resource allocation in underlay cognitive radio networks. There are different reinforcement learning techniques, with Q-learning being one of the most popular. Q-learning learns an action-value (also known as Q-value) function through the immediate reward feedback obtained from interacting with the environment. Because Q-learning is a type of gradual optimization process, it often suffers from the drawback of a slow convergence in finding the solution to the problem being studied. As a result, neural networks can be used to compensate for these Q-learning limitations in terms of generalization and function approximation capability. Deep Q-learning (DQL) is an emerging class of deep reinforcement learning algorithms which is capable of combining the process of reinforcement learning with a class of neural networks known as deep neural networks to approximate the Q action-value function. We deploy Q-learning and DQL as the learning methods to solve the optimization problems which will be set out in this chapter. This chapter consists of six sections; the first section reviews specific studies focused on resource allocation for heterogeneous underlay cognitive radio networks, in which not only the underlay DSA constraint but also the traffic-centric QoS/QoE constraint(s) are targeted to be met. In sections 3.4 and 3.5 we propose a single-layer and a cross-layer approach, respectively, where the physical layer transmission parameter(s) are learnt to be assigned such that not only the interference threshold of underlay DSA is satisfied but also the SN performance metric (i.e. MOS in this work) is maximized.

3.1 Literature Review

As stated in chapter 1, researches focused on resource allocation for underlay cognitive radio networks can be categorized into two groups of centralized and distributed approach. The algorithms in each groups are differentiated based on the problem objective and criteria meant to be optimized. The considered objectives may generally include: QoS/QoE such as throughput (sum rate) and delay, fairness and interference. The underlay spectrum sharing necessitates low power transmissions in SN such that the transmissions in SN appear as noise below the noise threshold (in fact, noise plus interference threshold) to the PU. Although appearing simple, the power allocation for SU should be carefully done since creating the noise floor for the PU above a particular level will immerse the PU signal in noise and make it undetectable to the PU receiver. The next most important consideration for underlay cognitive radio network, where users subscribe for different type of traffics, is to meet differentiated QoS requirements of different types of traffic.

With regard to the interference criteria, different interference mitigation techniques have been proposed in literature to manage the interference to the PU in underlay CRNs. They basically proposed techniques which make use of orthogonal transmission in time, frequency, space to coordinate access among users. Multiple access techniques of TDMA, FDMA, SDMA, CDMA are among the earliest techniques used to maintain orthogonal transmissions [97]. In particular, interference mitigation in frequency can be implemented by allocating different frequency bands to different users. Similarly, by assigning different time slots to different users, different codes to different users and different spaces to different users orthogonality in time, code, and space is achieved, respectively.

Performance of CDMA-based underlay cognitive radios has been studied in many works (such as [98] and references therein). Among these works, we review the ones which focus on QoS/QoE provision for SUs in a heterogeneous network. The work in [99] investigates the resource allocation for video streaming over underlay cognitive radio networks. The transmission rate and power along with the source rate at each secondary video session transmission are jointly optimized to provide QoS guarantee to the video streams. The authors assume that the encoded and compresses live video packets are stored in a M/M/1 queue, and scheduled on a First-In-First-Out (FIFO) order. The optimization problem is formulated into Geometric Programming and then is converted to a convex optimization problem such that, subject to the underlay DSA, queuing delay and packet loss rate (caused by transmission error and queue overflow) constraints, the sum of the reciprocals of the source rates of all secondary sessions are minimized. In [100] a cross-layer approach is utilized to address the problem of resource allocation to stream multimedia content by maximizing the sum peak-signal-to-noise ratio (PSNR) of video sessions. In [101] optimization problem was formulated as a capacity maximization problem to implement power allocation for SUs under QoS and peak power constraints for SUs.

In addition to CDMA, as stated, FDMA and TDMA can be also used as potential radio access technologies in underlay cognitive radio networks. In [102] authors considered a system with number of transmission links more than the available orthogonal time or frequency slots, and studied for such system the problem of scheduling and power control with SINR constraints. They used mixed-integer programming to solve the proposed optimization problem. The considered transmission links are divided into two sets of primary and secondary users where PUs are guaranteed to receive service, while SUs compete for the right to access to the time and/or frequency slots. The work in [103] studied quality-aware cross-layer resource allocation in CR networks to meet the QoS requirements for both real-time and non-real-time video applications. [104] proposed a resource allocation for OFDM-based CR networks with per-subcarrier power limit. The proposed scheme in [105] considered SUs subscribing video traffic over OFDM-based network and attempts to jointly optimize the bit rate, subcarrier and power allocation for each SU such that the total network rate increases, which consequently results in improved video quality. The authors in [106] have proposed cross-layer resource allocation for scalable multi-user video stream which maximizes SUs's overall PSNR while guarantees PU's interference limit under imperfect cross-channel gain information.

Moreover, power control and adaptive beamforming can be also used to mitigate the interference [107] in underlay CRNs. As stated the first priority in underlay CRNs is that the spectrum sharing process should guarantee protecting the primary network transmission. Therefore an interference threshold (called interference temperature as well) is usually set at primary receiver and should not be violated by SN. This can be done through robust power control techniques to protect the PU transmission while don't significantly degrade the SINR of the SUs. With regard to the multi-antenna or MIMO secondary system, however, this is not the only scheme which is used to mitigate interference. For instance, transmit beamforming can be used to protect primary transmission [108–110]. Some works also assumed a small secondary network deployed far away from the primary transmitter, such that the interference caused by SN to the primary transmission could be neglected. (e.g., [111, 112]).

To do power allocation, the works mentioned above only considered QoS performance of SUs, and don't account for end-user perceived quality assessment metrics. In contrast to QoS, as stated, QoE has become increasingly highlighted for data transmission, especially for media delivery. The most common method to measure QoE is the MOS, which Originally was adopted to assess human voice quality and then extended as an end user-centric quality assessment to other transmission services, such as file downloading and video streaming [113]. MOS is a metric which can unify all different classes of service in the optimization. [114] proposed A QoE-based channel allocation scheme for video transmission, but the authors only considered the scenario in which all SUs subscribe for video transmission service. Authors in [115] proposed a QoE-based admission control algorithm for video transmission over underlay DSA. However, they just considered one SU transmission link and the problem of resource allocation among multiple SUs transmitting concurrently different type of traffics has not been addressed. [116] considered the resource allocation problem for

SUs subscribing for different types of applications in the underlay CR network. This work's optimization problem is developed based on maximization of overall user satisfaction in SN. All schemes mentioned above are not applicable unless the SN has (perfect or imperfect) knowledge about the cross-channel gains between the secondary transmitter and the primary receiver (channel gains from the devices in the secondary network (SN) to the devices in the primary network).

In the literature, resource allocation algorithms are also classified according to the technique used for solving the problem which can be an optimization technique, heuristic, game theory or graph theory dependent. In this chapter, we develop our resource allocation problem as an optimization problem. In this context, reinforcement learning has been seen as an effective algorithm to solve optimization problems of the same type as the one at hand. In [117] Q-learning has been used to solve resource allocation problem which developed as a non-convex non-linear optimization problem. It maximizes the capacity of energy-harvesting OFDMA-based underlay CRNs by proposing an intelligent power allocation algorithm and under multiple constraints (i.e. PU interference threshold, minimum transmission rate, power and energy constraints). The authors showed the outperformance of their proposed algorithm over traditional water filling power allocation algorithm. The authors in [118] presented a cross-layer underlay CDMA-based cognitive radio scheme that uses Q-learning to jointly adapts transmit rate, source and channel coding rate for secondary users. They compared their results against a single (physical) layer approach based on the average end-to-end distortion. They integrated to their proposed algorithm the idea of transfer-learning to decrease the number of iterations needed for the q-learning algorithm to converge. The work in [119] proposed a deep reinforcement learning-based technique to jointly perform channel selection and power allocation for OFDMA-base underlay cognitive radio networks such that the spectrum efficiency was maximized. For the fixed number of PUs and SUs in the network, Q-learning was used as an alternative algorithm to justify the effectiveness of utilization of deep reinforcement learning to solve this work's considered optimization problem. But the scalability of the proposed approach when the size of the network is increased has not been studied. In [120], the authors presented a cross-layer routing algorithm for underlay DSA to improve the overall QoE for video traffic via learning the parameters from the physical and the network layer. The work in [121] adopted a variant of deep Q-learning to reduce the communication overhead for ad-hoc networks while guaranteeing the energy efficiency for the SUs.

We, in this chapter, formulate the resource allocation problem for heterogeneous underlay cognitive radio network as an optimization problem. First, we jointly optimize transmit rate and power of SUs in the network under SINR constraint of PU transmission. We adopt Q-learning and deep reinforcement learning to solve our developed optimization problem. Second, we develop a cross layer approach to not only satisfy the PU SINR constraint but the end-to-end video frame delivery delay constraint. In both scenarios our target is to maximize the average QoE in SN while meeting the considered constraints. The main difference between our work and all reviewed works in this section is that we consider multiple non-orthogonally transmitting SUs accessing the same spectrum band with PU. We don't use orthogonal code, space, frequency or time to give the users access to the common medium at the same time. We demonstrate that our proposed technique can achieve a lower queuing delay for the SUs subscribing for the video traffic, without violating the PU interference constraint. We then integrate transfer learning into our proposed technique to accelerate the performance of adopted reinforcement learning and identify the best practices to transfer learning between cognitive radios so as to reduce communication overhead and to identify the node from which to transfer knowledge. As mentioned earlier in this chapter, we consider a heterogeneous network consisting of SUs carrying either data or video type of traffic, and for this network we develop resource allocation solutions subject to some constraints. Before presenting the resource allocation solutions, we discuss the leveraged MOS models for considered types of traffic along with characteristics of assumed traffics.

3.2 MOS models

As stated, the QoE which represents the end-user centric quality assessment is gaining significant attention as we are moving toward 5G era. Consequently, we opted QoE as the network performance metric and specifically the MOS to model the QoE. In this chapter, the network is optimized based on average MOS across all data and video traffic sessions transmitted by SUs. In the following, we present the MOS formulas which are utilized in this chapter to quantify quality for delivered data and video traffic.

The MOS for data traffic is calculated based on the transmit rate, $T_i^{(s)}$, experienced by the end user, as follow [122]:

$$MOS_D = a \log_{10}(b\, T_i^{(s)}), \tag{3.1}$$

where Q_D is the data traffic MOS, while a and b are parameters calculated using the maximum and minimum data quality perceived by the end-user. If the transmit rate of a user is R and the receive rate is also R, then the packet loss is zero, and the quality perceived rate of end-user in terms of MOS should be maximum, that is 5. While MOS value of 1 is assigned to a minimum transmission rate.

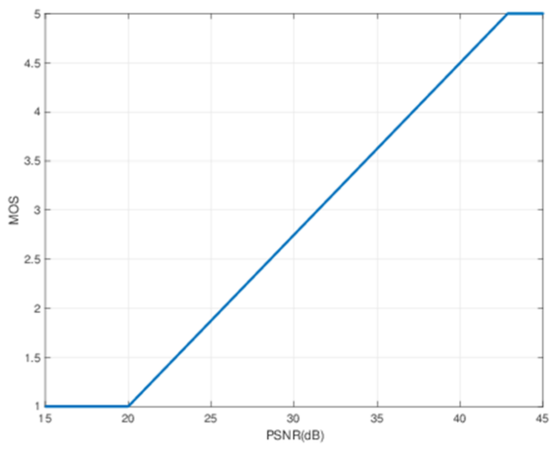

FIGURE 3.1: MOS versus PSNR.

As video quality assessment metric, peak signal-to-noise ratio (PSNR) is commonly accepted to objectively measure the coding performance of video. However, it is known that PSNR does not accurately reflect subjective human perception of video quality [123]. A wide variety of techniques have been proposed to estimate user satisfaction for video applications (a survey of video quality assessments can be found in [39]), among which [113] proposed a simple linear mapping between PSNR and MOS, as shown in Fig 3.1, which assigns MOS value of 4.5 for PSNR of 40 dB and MOS value of 1 for PSNR of 20 dB. The limits arise from the fact that received video sequences experiencing PSNR equal to 40 dB are almost indistinguishable from the transmitted one and those below 20 dB exhibit very severe quality degradation.

PSNR[dB]	MOS
>37	5 (Excellent)
31-37	4 (Good)
25-31	3 (Fair)
20-25	2 (Poor)
<20	1 (Bad)

TABLE 3.1: Possible PSNR to MOS Conversion [2]

The work in [2] presented a heuristic mapping from PSNR to MOS as shown in Table 3.1, while, according to the recommendation ITU-R BT.500-13 [124], the relationship between MOS and an objective measure of picture distortion have a sigmoid shape. Consequently, [125] claimed that if the picture distortion is measured with an objective metric, e.g., PSNR (dB), then a logistic function can be used to characterize the relation between MOS and PSNR, as follows:

$$MOS_V = \frac{a}{1 + exp(b\ (PSNR - c))}, \tag{3.2}$$

where MOS_V denotes the MOS for video, and a, b and c are the parameters of the logistic function. In this chapter the logistic function was selected to model the quality of video traffic. To compute the parameters of the logistic function (3.2), an array of PSNR and corresponding MOS values is needed. Since the PSNR of reconstructed video changes as a function of bit rate with the characteristics of the video sequence itself, we averaged over the PSNR-bit rate functions for multiple MPEG-4 coded video sequences at different resolutions, 240p, 360p and 480p and obtained one average PSNR-bit rate curve. The video sequences were combined in the respective proportions 39%, 32% and 28% , same as the proportions used in [126]. The video sequences used were "Flowervase" and "Race Horses", at 30 frames per second (fps) and resolution 480p, while for resolution 360p "Tennis" and "Park Scene" at 24 fps were selected. As a result, it was observed that a function of the form $PSNR = k \log T_i^{(s)} + p$ can be used to very closely approximate the average PSNR-bit rate curve, where q_V is the video quality measured in PSNR, $T_i^{(s)}$ is the video transmission bit rate and k and p are constants. To get the parameters of the logistic function formula as in (3.2) an array of PSNR values and its corresponding MOS values is needed. In this report, we first computed PSNR values for an array of video bit rate (which itself computed through equation (**??**) for all candidate actions) using combined PSNR-bit rate curve. To get MOS values corresponding to the computed PSNR values, we used the linear mapping to map PSNR to MOS. We also used table-based conversion to obtain another array of MOS values for the same PSNR values. Then we averaged the two MOS arrays to obtain final MOS array. After obtaining the MOS value corresponding to the PSNR values, the parameters for the logistic function were found to be c = 6.6431, d = 0.1344 and f = 30.4264.

3.3 Traffic Models

In this chapter it is assumed that an active data session user runs best effort services. For best effort users the network does not guarantee any certain QoS level. However, for the video traffic an end-to-end delay constraint needs to be satisfied (in this section we don't take into account the delay constraint, this constraint will be integrated in the technique through a cross-layer approach which is focus of the next section). Detailed description of the traffic models is presented as follow:

3.3.1 Best Effort Traffic Model

As the model of best effort traffic we consider FTP (File Transfer Protocol) with the model from [3]. The parameters of the model are the size of the transferred files and their separating reading times which follow the truncated Lognormal distribution and the Exponential distribution, respectively.

3.3.2 Video Streaming Traffic Model

Our video streaming traffic model follows [4] where each video frame has a fixed number of packets (P packets). Each packet arrives at a random

TABLE 3.2: Parameters of FTP traffic model [3]

Parameters	Statistical characteristic
File size	Truncated Lognormal distribution $\mu = 2$ Mbytes, $\sigma = 0.722$ Mbytes, maximum file size = 5 Mbytes, PDF: $$f_x = \frac{1}{\sqrt{2}\sigma x} e^{(}\frac{-(\ln x - \mu)^2}{2\sigma^2}), x > 0 \quad (3.3)$$
Reading time	Exponential distribution mean = 180 seconds, PDF: $$f_x = \lambda e^{-\lambda x}, x > 0 \quad (3.4)$$

time interval while the inter-arrival time for each frame is deterministic. To model the size and the inter arrival time of packets in a frame, the truncated Pareto distribution is used. Parameters of video streaming traffic are given in table 3.3.

3.4 Single Layer Resource Allocation Leveraging Reinforcement Learning

In this section we explain our resource allocation algorithm for underlay CRN in which a CE embedded in each CR only has two inputs; one being as the learned information from the effect of the previous actions on the wireless environment. Under the assumption of link adaptation, this input helps the SUs to infer the state of the primary link and then be able to maintain the interference to the PN below the preset tolerable threshold. This information can be achieved through the technique presented in chapter 2, [127]. The second input to the CE is the information achieved from its own feedback channel. We assume that all transmissions in the network adopt adaptive modulation and coding (AMC) at the physical layer. Thus, each CE's second input is its own transmission link AMC feedback channel which represents the attainable bit rates (and later we will see that it is

TABLE 3.3: Parameters of video streaming traffic model [4]

Parameters	Statistical characteristic
Inter-arrival time between beginning of successive frames	Deterministic(100 ms for 10 fps, 50 ms for 20 fps)
Number of packets in a frame	Deterministic (8 packets)
Packet size	Truncated Pareto distribution mean = 100 bytes, maximum = 250 bytes, minimum = 53 bytes, k = 50 bytes, a=1.2 PDF: $f_x = \frac{ak^a}{x^{a+1}}, x > 0 \quad (3.5)$
Inter-arrival time between packets in a frame	Truncated Pareto distribution mean= 6ms, maximum = 12.5 ms, minimum= 2.5 ms, k= 2.5 ms, a=1.5 PDF: $f_x = \frac{ak^a}{x^{a+1}}, x > 0 \quad (3.6)$

used to compute the reward for the RL algorithm). The reason of naming the upcoming technique as a single layer resource allocation technique is that it incorporates the input only from the physical layer while in the next section the presented algorithm incorporates the input from the data link layer as well.

3.4.1 System Model

We study an underlay DSA where a PN with a single primary link coexists with a SN consisting of N SU transmission links. Each SU communicates simultaneously with its respective secondary base station (SBS) in a frequency band shared with the PN under underlay DSA paradigm. The PN is incumbent to the spectrum band being used, so SUs need to meet the interference constraint associated with underlay DSA by adapting their transmit power accordingly. We assume that all transmissions in the network adopt adaptive modulation and coding (AMC) at the physical layer. To incorporate the underlay DSA constraint into the algorithm, a preset SINR threshold β_0 is considered to be met at the primary base station (PBS) as:

$$SINR^{(p)} = \frac{G_0^{(p)} P_0}{\sigma^2 + \sum_{j=1}^{N} G_j^{(p)} P_j} \geq \beta_0, \tag{3.7}$$

where P_0 is the PU transmit power, P_j is SU$_j$'s transmit power, $G_0^{(p)}$ and $G_j^{(p)}$ are the PU to PBS channel gain and the jth SU's channel gain to the PBS, respectively. Background noise power is denoted by σ^2. In order to also consider the QoE of the received traffic in the SN, we define the SINR for the ith SU at its corresponding SBS as:

$$SINR_i^{(s)} = \frac{G_i^{(s)} P_i}{\sigma^2 + G_0^{(s)} P_0 + \sum_{j \neq i} G_j^{(s)} P_j}, \tag{3.8}$$

where $G_0^{(s)}$ is the channel gain between the PU and SBS, P_i is the ith SU's transmit power, and $G_i^{(s)}$ is the ith SU's channel gain. In order to incorporate into the algorithm the QoE of the received traffic in the SN, we impose an additional SINR requirement β_i needed to be met at ith SBS. The two SINR requirements, for the PN and the SN, can be expressed as,

$$\begin{cases} SINR^{(p)} \geq \beta_0 \\ SINR_i^{(s)} \geq \beta_i, \quad i = 1, \cdots, N. \end{cases} \tag{3.9}$$

In this work we assume that the use of AMC on all links can be leveraged to perfectly estimate all needed channel gains through an active learning-based techniques [128]. If the SINR constraints for both secondary and primary networks in (3.9) are met with equality, the solution to the power allocation for each SU can be achieved as,

$$P_i = \frac{\Psi_i(\sigma^2 + G_0^{(s)} P_0)}{G_i^{(s)}(1 - \sum_{j=1}^{N} \Psi_j)}, \quad i = 1, 2, ..., N, \tag{3.10}$$

where $\Psi_i = (1 + \frac{1}{\beta_i})^{-1}$. The condition $1 - \sum_{i=1}^{N} \Psi_i > 0$ needs to be satisfied to ensure that the power allocation is valid. After replacing the SU powers obtained from (3.10) in (3.7) and (3.8) , (3.9) can be rewritten as,

$$\textstyle\sum_{j=1}^{N} \alpha_j \Psi_j \leq 1, \tag{3.11}$$

where

$$\alpha_j = \frac{G_j^{(p)}(\sigma^2 + G_0^{(s)} P_0)}{G_j^{(s)}(G_0^{(p)} P_0/\beta_0 - \sigma^2)} + 1, \tag{3.12}$$

Following the use of AMC on all links, the relation between transmit bit rate and the SU threshold SINR (β_i) follows (2.20) [129]. Referring to (3.9) and (3.10) we can see that β_i needs to be adjusted at each SU such that the underlay DSA interference requirement is met and SUs QoE is also guaranteed. According to (3.10) and (2.20) this adjustment is applied by adapting transmit power and rate. On the other hand, the performance metric in the SN and its increase needs to be also taken into account. We Incorporated this into our algorithm by defining the optimization problem based on maximizing the SN's performance metric. Specifically, we defined our goal to be maximizing the network performance metric while satisfying a total interference constraint to the PU. Based on the mentioned property of MOS which provides a common and uniform measurement scale for all type of traffics including video and data, to compute the average MOS for a network with an heterogeneous traffic we are allowed to add MOS for all video and data sessions, as follow: $\frac{1}{N}(\sum_{i=1}^{U} MOS_D + \sum_{i=U+1}^{N} MOS_V)$, where U is the number of SUs transmitting data while the remaining $N-U$ users transmit streaming video. As such, the optimization problem for our single layer underlay DSA scheme is to maximize the network average MOS while satisfying the underlay DSA interference constraint:

$$\begin{aligned} \{(\hat{\beta}_i)\} = \arg\max_{\beta_i} \tfrac{1}{N} \textstyle\sum_{i=1}^{N} MOS_{(DorV)}^{(i)}(\beta_i), \\ \text{s.t.} \quad \textstyle\sum_{i=1}^{N} \Psi_i(\beta_i) \leq 1, \\ \textstyle\sum_{j=1}^{N} \alpha_j \Psi_j(\beta_j) \leq 1, \end{aligned} \tag{3.13}$$

This optimization (resource allocation) problem is solved through discrete-time Markov decision process (DTMDP) modeling and the use of a reinforcement learning (RL). The RL learning [23] has been shown as an effective solution for the resource allocation in communication networks. The RL agent can generate near optimal solutions through an immediate reward achieved from interactions with the environment. Through optimizing the current reward, the RL agent achieves a long-term optimizing goal, which is important for dynamic systems such as wireless networks. We consider two variations of RL including the Q-learning (as the most popular RL algorithm) and the deep Q-network (DQL) (as an emerging class of RL algorithms which combines RL with neural networks). In the next sections we explain how we deploy mentioned RL algorithms to solve the optimization problem in (3.13).

3.4.2 Q Learning-Based Cognitive Engine

The Q-learning algorithm (also called table-based standard Q-learning algorithm) deployment [25] requires the definition of a set of states S, a set of actions A and a reward function. The reward function represents the effect of a selected action on the environment from which each of the learning agents (CRs/SUs) will be guide to choose the next action from set of actions. The same set of actions as well as states are assumed for all SUs. Each SU conducts a search into the finite discrete space of candidate target SINR, denoted by $\mathcal{A}^{(i)} = \{\beta_1^{(i)}, \cdots, \beta_n^{(i)}\}$ to find the optimal solution which not only meet the constraints in (3.9) but results in the better average MOS for SN. The Q-Learning algorithm considers the environment as a finite-state, discrete-time stochastic dynamical system. The learning agent observes its current state $s \in S$ and accordingly take an action $\pi(s) \in \mathcal{A}$, under a certain policy π, which involves scalar immediate reward $R_t^{(i)}$. The problem then is to find a policy which maximizes the received discounted reward V with a discount factor γ $(0 < \gamma < 1)$, [25]. At the same time with choosing one strategy from $\mathcal{A}^{(i)}$, each SU adapts its transmit power and rate through (3.10) and (2.20), then observes the changes in the system and its own transmission as well. Each SUs will seek to find an optimal policy to maximize its own MOS while the SINR constraints (3.9) are satisfied.

3.4.2.1 State Space

The states, $S_t = (I_t, L_t)$, are defined to reflect the interference caused by the SUs where,

$$I_t = \begin{cases} 0, & \text{if } \sum_{i=1}^{N} \Psi_i(\beta_t^{(i)}) < 1 \\ 1, & \text{otherwise,} \end{cases} \tag{3.14}$$

$$L_t = \begin{cases} 0, & \text{if } \sum_{i=1}^{N} \alpha_i \Psi_i(\beta_t^{(i)}) \leq 1 \\ 1, & \text{otherwise.} \end{cases} \tag{3.15}$$

3.4.2.2 Reward

The reward function is also defined as a function of the state and the local action,

$$r_t^{(i)}(a_t, s_t) = \begin{cases} M, & \text{if } I_{t+1} + L_{t+1} > 0 \\ MOS_{(DorV)}^{(i)}, & \text{otherwise,} \end{cases} \tag{3.16}$$

with the assumption of M being a constant smaller than the reward of any other strategies in exchange of taken an unsuccessful action resulting in interference constraints violation (3.9). While in the case of satisfying the interference constraints, MOS of the received traffic which is either video (MOS_V) or data (MOS_D) is considered as the immediate reward.

We also assume that the SUs do not know the other's action or the effect of joint actions on states and considers the others as part of the environment. Then the SUs repeatedly make their decisions and finally obtain their optimal policies to maximize the expected sum of discounted reward as expressed in equation (1.2).

According to Bellman's principle of optimality [26], the solution to (1.2) can be obtained by taking the optimal action if all the strategies thereafter are optimal. The optimal action can be approached by the Q-function, which is updated as in (1.4). Consequently, according to (3.16) and (1.3) each SU repeatedly take actions to maximize its individual experienced MOS, which results in maximization of the average SN's MOS and leads to finding a solution for (3.13).

Algorithm 1 summarizes the steps needs to be taken by each SU in order to implement the individual learning mechanism. It should be noted that because of the initialization for all the SUs $Q_0^i = 0$, the algorithm will first perform an initial exploration phase, where each Q-table entry is visited once, then will through an exploitation phase. By adjusting β_i, and consequently corresponding $T_i^{(s)}$, the SU tries to obtain an optimal power assignment in order to not only satisfies the interference threshold in (3.9) but maximize the network performance metric (i.e. average MOS in SN).

Algorithm 1 Multi-agent Q learning-based learning framework

```
Initialization: $Q_0 = 0$ for all the SUs
    for time $t < t_{\max}$ do
        for all SU$_i$, $i = 1, \cdots, N$ do
            Select the action $a_t^{(i)} = \arg\max_{a_t^{(i)}} Q(s_t^{(i)}, a_t^{(i)})$
            Update the state $s_{t+1}^{(i)}$ (3.19), (3.20) and the reward $r_t^{(i)}$, (1.3).
            Update Q-value $Q_t^{(i)}(s_t, a_t)$, (1.4).
        end for
    end for
```

3.4.3 DQL Learning-Based Cognitive Engine

Equation (1.2) is a value iteration algorithm that converges to the optimal action-value function if $t \to \infty$. This approach is impractical. Consequently, a function estimator is used to estimate the optimal action-value function. In a DQL, a neural network has been proposed as an efficient nonlinear approximator to estimate action-value function $Q_i(s, a; \theta_i) \approx Q_i^*(s, a)$, [16]. In this paper we used a fully connected feed-forward Multilayer Perceptron (MLP) network to approximate the action-value function. DQL takes advantage of neural network as a powerful non-linear function approximator to estimate the action-value function. To improve the learning performance, we include a technique known as "experience replay" to the DQL algorithm. In experience replay, at each time step, the experience of each agent with the environment is stored as a tuple $e_i(t) = (a_i(t), s_i(t), T_i(t), s_i(t+1))$ into a replay memory. The replay memory of ith. agent is denoted as $D_i(t) = e_i(1), \ldots, e_i(t)$. Moreover, each agent utilizes two separate MLP networks as Q-network approximators:

one as action-value function approximator $Q_i(s, a; \theta_i)$ and another as target action-value function approximator $\hat{Q}_i(s, a; \theta_i^-)$, where θ_i and θ_i^- denote the current and old parameters respectively. At each time step, the current parameters θ_i of each agent's action-value function are updated through a mini-batch of random samples of transitions $(a_i, s_i, T_i, \hat{s}_i)$ from the display memory D_i. The old parameters θ_i^- of the target action-value function are replaced by the updated parameters θ_i of action-value function at every C iterations. The parameters of θ_i action-value function are updated by utilizing a gradient descent algorithm based on the following cost function:

$$L(\theta_i) = E[(r_T^i(s, a) + \gamma \max_{\hat{a} \in A}(\hat{Q}_i(\hat{s}, \hat{a}; \theta_i^-)) - Q_i(s, a; \theta_i))^2] \tag{3.17}$$

algorithm 1 summarizes the steps that each SU needs to take in order to implement the DQL mechanism. It should be noted that the action selection procedure follows the ϵ-greedy policy which means that an action is randomly chosen from the action set $\mathcal{A}$ with probability ϵ, otherwise an action with the highest output to the action-value function is chosen.

Algorithm 2 Multi-agent DQL-based learning framework [16]

```
for all $\text{SU}_i$, $i = 1, \cdots, N$ do
    Initialize replay memory
    Initialization of the neural network for action-value function $Q_i$ with random weights $\theta_i$
    Initialization of the neural network target action-value function $\hat{Q}_i$ with $\theta_i^- = \theta_i$
end for
for $t < T$ do
    for all $\text{SU}_i$, $i = 1, \cdots, N$ do
        Select a random action with probability $\epsilon$.
        Otherwise select the action $a_t^{(i)} = \arg\max_{a_t^{(i)}} Q_i(s_t^{(i)}, a_t^{(i)}; \theta_i)$.
        Update the state $s_{t+1}^{(i)}$ ((3.19), (3.20), and the reward $r_t^{(i)}$, (3.16).
        Store $e_i(t) = (a_t^{(i)}, s_t^{(i)}, r_t^{(i)}, s_{(t+1)}^{(i)})$ in experience replay memory of $\text{SU}_i$, $D_i$.
        Update parameters ($\theta$) of action-value function $Q(s_t^{(i)}, a_t^{(i)}; \theta_i)$, by sampling random mini-batch of transitions from $D_i$.
        every C step update parameters of target action-value function $\theta_i^- = \theta_i$
    end for
end for
```

3.5 Cross Layer Resource Allocation Leveraging DQL

Being an end-to-end performance metric, the QoE depends on transmission parameters from multiple layers of the protocol stack. In the case of video streaming, the end-user QoE is primarily determined by the QoS parameters such as the connection effective bit rate, delay and delay jitter. Nevertheless, meeting video QoE goals is difficult due to the limited transmit bandwidth, dynamic channel characteristics and of the video content itself. On the other hand, authors in [130] presented a situation where PSNR is not a reliable video quality metric to be used to model the subjective QoE scores. They provide evidence that PSNR is inaccurate in measuring video quality of a video content encoded at different frame rates. Therefore, we decided to incorporate video frame rate and accordingly video frame delay constraint in our resource allocation algorithm and to later examine the adaptability of our proposed algorithm to the changes in video frame rate. Therefor, regarding the resource allocation algorithm being presented in this section, to meet QoE goals it is necessary to also consider the delay from the data link layer scheduler's queue. Since in practice, the scheduler's queue length (buffer size) is finite, when the buffer is full and overflow occurs, excess packets have to be dropped that results in packet loss and reduction in the QoE. Moreover, packet loss is not only caused by queuing overflow at the link layer scheduler but also by the transmission errors at the physical layer. To mitigate packet transmission errors, hybrid automatic repeat request (HARQ) coupled with AMC is widely used, but it introduces higher queuing delay. The increased delay is caused because the packets are required to be kept in the buffer until successful delivery or till a maximum number of unsuccessful re-transmission attempts have been reached. At the same time, the link bit rate affects the QoE directly by influencing the queuing delay. Thus, to achieve the best QoE, it is important to follow a cross-layer approach that jointly considers delivery delay (inherently involving queuing delay) and bit rate. Our goal in this section is to develop a cross-layer resource allocation algorithm for underlay DSA that aims at maximizing the overall QoE (which involve minimizing the video frame delivery delay) across transmission of all video or data traffics while satisfying the underlay DSA interference constraint.

3.5.1 System Model

Since we implement underlay DSA, all equations from (3.7) to (3.12) are applied in this section as well. Moreover, all transmissions deploy AMC technique to adapt the transmit rate to the state of the channel (as a result, (2.20) is applicable here as well). Each SU data link scheduler (which queues arriving packets) is equipped with a first-in-first-out (FIFO) finite length buffer, followed by a CE which includes AMC and transmit power controller (as depicted in Fig. 3.2).

The AMC controller devises the state of the link (in terms of SINR) through the information conveyed on the feedback channel from which the attainable bit rate and accordingly the quality of delivered traffic can be estimated. We also assume that the implemented HARQ operates based on packet combining and ACK/NACK feedback before proceeding to the next transmission. Therefore, while HARQ improves transmit error performance, it

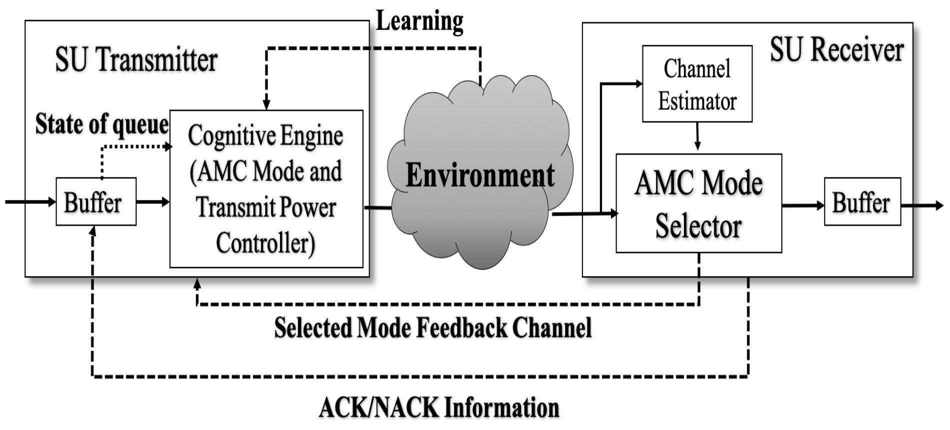

Figure 3.2: System model overview.

also increases the average packet delivery delay. In our system setup we assume that each packet can be re-transmitted at most three times.

In our proposed scheme in this section the resources, which need to be allocated for each SU, are transmit power and rate and are determined by the CE. These decisions are made based on the information obtained from the three inputs to the CE. The first two inputs contain the information about the interference created to the PN (which can be inferred through the technigue presented in chapter 2 [127]) and the scheduler's buffer status (which affects the experienced QoE). The AMC feedback channel information input to the CE also represents attainable bit rates and used to compute the part of total reward for RL algorithm.

With regard to the video traffic, as mentioned, the transmit parameter setting concerns with not only meeting the underlay DSA constraint but also satisfying the end-to-end video frame delay requirement. In order to meet the real-time/streaming video traffic's delay constraint, video packets in the queue may need to be served at a larger rate. As such, the transmit rate needs to be adapted to the the state of the queue and specifically to the video frame experienced delay. Therefore, the optimization problem for our underlay DSA scheme is to maximize the network average QoE while satisfying the delay and interference constraints, as follow:

$$\begin{aligned} \{(\hat{\beta}_i)\} = \arg\max_{\beta_i} \tfrac{1}{N} \textstyle\sum_{i=1}^{N} Q^{(i)}_{(DorV)}(\beta_i), \\ \text{s.t.} \quad \textstyle\sum_{i=1}^{N} \Psi_i(\beta_i) \leq 1, \\ \textstyle\sum_{j=1}^{N} \alpha_j \Psi_j(\beta_j) \leq 1, \\ D_i \leq \breve{D}_i \quad (i = 1, \ldots, N). \end{aligned} \tag{3.18}$$

where D_i represents the experienced end-to-end frame delay for SU$_i$'s traffic and $\breve{D}_i$ denotes its corresponding end-to-end frame delay requirement. Thus each SU selects its target SINR β_i and adjust its transmit parameters $T_i^{(s)}$ and P_i, with the goal of maximizing its own delivered traffic QoE, which leads to the maximization of average experienced QoE in the SN.

3.5.2 DQL-Based Learning Framework

In solving the problem (3.18), we want the CE to be able to learn the underlying characteristics for the traffic and also for the wireless environment. We decided to use a RL algorithm to run the cognition part of the CE, as it is able to gradually learn the effect of selected actions on the wireless environment in general and on the conveyed traffic in particular through the immediate reward, and results in the best action to be taken using the expected reward. Furthermore, due to its state-of-the-art learning performance we choose to implement RL using the DQL framework. As such, each SU in the network will be equipped with a single RL agent which at time t takes an action $a(t) \in \mathcal{A} = \{a_1, a_2, \ldots, a_{|\mathcal{A}|}\}$ while the environment is in state $s(t) \in \mathcal{S} = \{s_1, s_2 \ldots, s_{|\mathcal{S}|}\}$. During this interaction, the agent achieves an immediate reward $r(s, a)$ and the system transitions into a new state $s(t+1) \in S$. The immediate reward represents the effect of the selected action on the environment and guides each of the learning agents to choose the next action from $\mathcal{A}$.

3.5.2.1 State Space

Let the state of the SU$_i$'s queue at time t be O_t^i and the measured waiting time for the last transmitted P packets in the queue be M_t^i. Since the QoE is affected by variables from the multiple layers of the network protocol stack, the environment seen by the CE also needs to encompass these multiple layers. This concept is incorporated in the following definition of the states $S_t = (I_t, L_t, O_t^i)$:

$$I_t = \begin{cases} 0, & \text{if } \sum_{i=1}^{N} \Psi_i(\beta_t^{(i)}) < 1 \\ 1, & \text{otherwise,} \end{cases} \tag{3.19}$$

$$L_t = \begin{cases} 0, & \text{if } \sum_{i=1}^{N} \alpha_i \Psi_i(\beta_t^{(i)}) \leq 1 \\ 1, & \text{otherwise.} \end{cases} \tag{3.20}$$

$$O_t^i = \begin{cases} 0, & \text{if } M_t^i \leq \breve{D}_i \\ 1, & \text{otherwise.} \end{cases} \tag{3.21}$$

This definition reflects that the state of the SU$_i$'s queue O_t^i at time t will be 0 if M_t^i meets the end-to-end frame delay requirement of the traffic (we recognize that channel and decoding delays are comparably negligible). Note that there is no end-to-end delay requirement corresponding to the best-effort traffic that means for the data traffic O_t^i is 0.

3.5.2.2 Reward

We defined a total reward function as the weighted sum of two rewards, r_1 and r_2. The reward r_1 represents the contribution from the queuing delay and takes a positive value when the last P transmitted packets are serviced within their delay requirements. Fig. 3.3 shows the reward r_1 function for a 10 fps video source with a 150 ms end-to-end frame delay requirements. This function is updated according to the change in the video source frame rate and its delay requirement. For data traffic, r_1 takes a constant positive value.

The second component in the total reward function r_2, incorporates the effect of transmit rate on the QoE and is defined as,

$$r_2^{(i)}(a_t, s_t) = MOS_{(DorV)}^{(i)}, \tag{3.22}$$

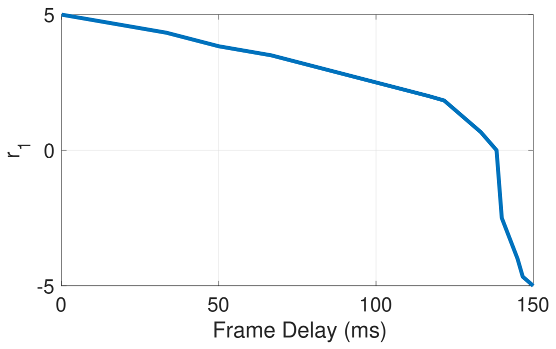

FIGURE 3.3: Reward function for video frame delay.

where Q_V and Q_D are the delivered video and data MOS, respectively. Finally, the total reward function is defined as,

$$r_T^i = \begin{cases} J, & \text{if } I_{t+1} + L_{t+1} + O_{t+1}^i > 0 \\ w_1 r_1^i(a_t, s_t) + w_2 r_2^i(a_t, s_t), & \text{otherwise,} \end{cases}$$

with the assumption of J being a negative constant, in exchange of taken an unsuccessful action resulting in interference and delay constraint violation (3.21). For simplicity we consider w_1 and w_2 equal to 1 [131].

3.6 Integration of Transfer Learning into Reinforcement Learning

As mentioned the cognition capability of CE can be achieved by RL, a technique that has been widely used in this context. This technique presents the appeal that allows the CR to learn without prior knowledge of the system model, the best set of action (policy) at each environment state. This is achieved by following an iterative process of performing trial actions and observing their effect to reward the successful decisions and penalize the failed ones. However, the long learning time involved in the RL iterative process motivated researchers to explore the concept of transfer learning, originally proposed as a single-agent technique to transfer experience from one task to another similar task [132]. However, the idea of transfer learning can be extended to multiple agents, specially in the case of a wireless communication networks where agents have inherent capabilities to share information.

The philosophy behind transfer learning is to allow well-established and expert cognitive agents (i.e. base stations or mobile stations in the context of wireless communications) to teach newly activated and naive agents. This exchange of learned information is used to improve the performance of a distributed CR network. With regard to the DQL algorithm a Q action-value function is estimated (learnt) iteratively based on the reward for each action. Each cognitive SU first learns about its surrounding environment by running the DQL learning algorithm and updating the parameters of its Q action-value function based on the received immediate reward. Therefore, the Q action-value function and specifically its parameters will reflect the effects of the actions on the wireless environment. Because part of the wireless environment involves the interference created by each SU to the rest of the system, the parameters will reflect both the individual local wireless environment for each SU and the collective interrelation between the system components. With regard to the standard table-based Q-learning the entries of the Q-table will carry the information regarding the effect of actions on the environment. When a SU joins the already learnt system, the environment shows limited changes, therefore it is inefficient to re-run the cognitive cycle and disregard the awareness of the environment captured by other SUs already in system. Hence, this awareness of the environment which is reflected in Q action-value function parameters (or Q-table entries for table-based Q-learning) can be taught to the new joined SU in order to decrease the learning time and also improve the learning performance.

Nevertheless, the application of transfer learning introduces new research questions, namely whom to obtain the knowledge from, how much information to exchange and when to stop the exchange of information. We in this work explore the first two questions. To answer the first question of whom to obtain the knowledge from we have run an extensive simulations and explored the similarity between the Q action-value function parameters of the SUs in the system. Fig. 3.4 shows a comparison between parameters (θ) of the action-value function of SUs in terms of their mean square error (MSE). Results are presented in the form of frequency values for different distances among SUs. We selected one user as a reference user, and showed different distances by the first nearest SU to the reference one, the second nearest user to the reference one and the farthest one. The comparison was done over 4000 runs of the DQL learning algorithm for a network with 4

SUs, each run corresponding to a different random location of SUs. Further setup details are described in Section 3.7. It can be seen that the probability of having MSE equal to 0.5 between the reference SU and the first nearest SU is more than 70% while it is less than 40% for the farthest.

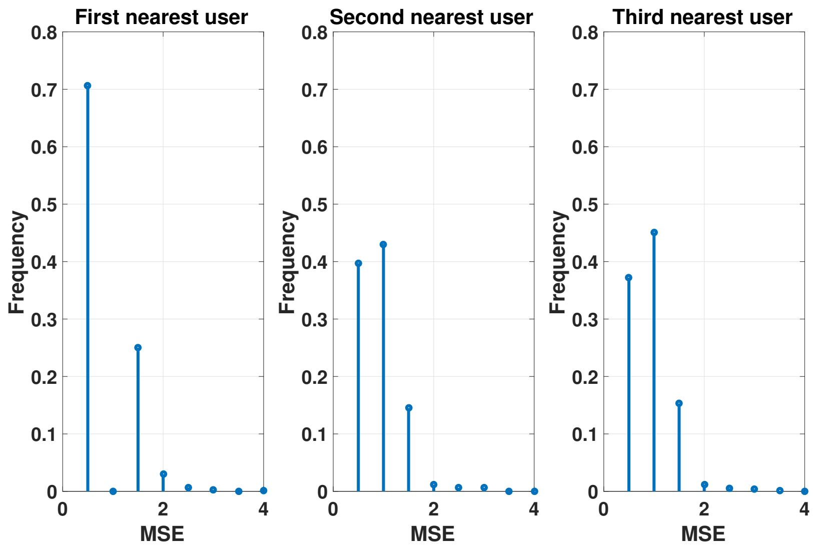

FIGURE 3.4: Distribution of the MSE of the action-value function parameters (θ) between a first SU and other SUs in a secondary network of 4 users.

Fig. 3.4 also shows that as the distance between SUs increases, the rate of similarity (MSE = 0.5) between the parameters decreases and reaches to less than 40% for the farthest user to the reference user. As a result, this figure shows the somewhat expected result that parameters of the Q action-value function become more similar as SUs become closer. The reason for this behavior is that the parameters of Q function not only incorporate information about each SU's local environment achieved by immediate reward, but also about the interaction of all system components.

As a result, we implement transfer learning where SUs already operating in the network initialize their cognitive cycle with their own action-value function already learnt through algorithm 2 and the newcomer SU initializes parameters of its action-value function with the parameters of the nearest node. In the next section we examine the effect of transfer learning on the learning performance. The mentioned transfer learning mechanism is

Algorithm 3 Integration of Transfer Learning into RL

```
(Run as a new SU joins the network)
Add one new SU as $\mathrm{SU}_{N+1}$
Assign the nearest node $n(i)$ to the new SU
Initialize $Q_{(N+1)}$ with parameters of the action-value function of its nearest neighbor $\theta_{(N+1)} = \theta_{(n(i))}$ ($Q_{(N+1)} = Q_{n(i)}$ regarding the table-based Q-learning).
for all $\mathrm{SU}_i$, $i = 1, \cdots, N+1$ do
    Restart individual learning algorithm 2 (algorithm 1 regarding table-based Q-learning) with the existing $N+1$ action-value function.
end for
```

shown in algorithm 3. Moreover, a low bit rate control channel is assumed as an indication if a newcomer (less experienced node) joins the network.

To reduce the communication overhead and find the answer to the question of how much information to exchange, it is worth exploring the effect of partial transfer of the nearest SU's action-value function parameters. To explore this option, we considered to take a closer look at the parameters on a layer by layer basis. Based on 2000 Monte Carlo simulation runs for a SN with 3 SUs, Fig. 3.5 shows for the same mean distance the distribution of the mean cross-correlation coefficients between the parameters of different layers of the action-value function of a video reference SU and its nearest user, at convergence point of the DQL algorithm. Fig 3.5 on top shows the distribution of cross correlation coefficients between two users's first layer parameters (i.e. the weights between the input and first hidden layer). It can be seen that for around 80% of the time the cross-correlation coefficients are between -0.1 and 0.1 , while for the second layer parameters (weights between first and second hidden layer) the span for more frequent cross-correlation coefficients increases and becomes between -0.1 and 0.3. With regard to the third layer (the weights between the second hidden layer and the output layer), Fig. 3.5 on the bottom shows that for around 80% of the time the cross-correlation coefficients between the last layer parameters of two users lie between 0.2 and 0.8. This figure clearly shows that the higher correlations belong to the last layer parameters and suggest an strategy that will be adopted next to significantly reduce the communication overhead when implementing transfer learning. Moreover, by illustrating a shift from symmetric distribution of negative and positive cross-correlation coefficients (seen for the first layer parameters) and a shift

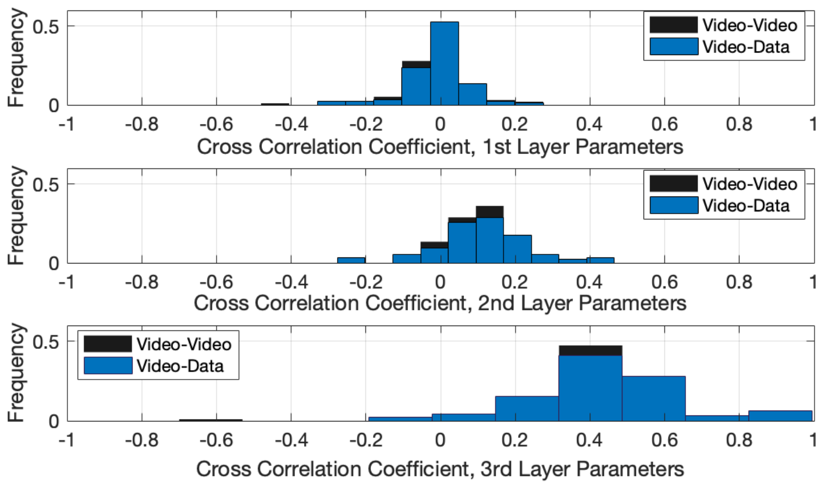

FIGURE 3.5: Distribution of the cross correlation between the action-value function parameters of the video reference SU and its nearest neighboring SU.

from negative cross-correlations to positive, this figure shows that the last layer parameters of two users tend to have stronger linear relation. Therefore, by exchanging only the third layer parameters (which are 3 compared to the entire 33 parameters) the communication overhead can be decreased by 90%. Moreover, this figure shows an additional interesting fact that the traffic type of the nearest SU does not significantly affect the distribution. Specifically, the traffic type of the nearest SU does not play an important role in better transferring the learnt information. Leveraging the observations in the previous paragraph, in this paper we consider the scenario in which new joined SU learns from the nearest SU and initializes its last layer action-value function parameters using the last layer action-value function parameters of the nearest SU, while randomly initializes the remaining two layers. Note that a control channel is assumed to exchange the parameters.

3.7 Results

In this section we evaluate the performance of our proposed single-layer and cross-layer resource allocation technique based on Monte-Carlo simulations. First, we consider our proposed single layer approach and show the results of its implementation on solving the optimization problem discussed in (3.13).

We adopt Q-learning along with DQL to not only find the solution to this problem, but through extensive simulation runs compare the performance of these two RL-based algorithms. We assume a primary network which consists of one primary link accessing a single channel. The target SINR for the PU is set at 1dB. The Gaussian noise power and the transmit power of PU are set to be -90 dB and 23dBm, respectively. All SUs and PUs are distributed randomly around their respective base stations within a circle of radius 300m. The secondary base stations are also distributed randomly at a distance from the primary base station chosen with a uniform distribution between 10m and 700m. Channel gains follow a log-distance path loss model with path loss exponent equal to 3. For a single SU, its SINR could be chosen from the finite discrete set $\{-9, -5, -1, 3\}$ dB. Regarding to the learning algorithm, the same learning rate $\alpha = 0.01$ and discounting factor $\gamma = 0.9$ are assumed for all SUs. In the ϵ-greedy exploration, ϵ is initially set to be 0.8, as the number of iteration increases it converges to 0.

As action-value function and target action-value function approximators we used two separate feed forward neural networks for each SU. Each neural network equipped with two fully connected hidden layers with 3 and 2 neurons. The capacity of replay memory and mini-patch was set to 100 and 10, respectively. The number of Monte-Carlo simulations was set to $K = 500$, and the results are obtained by averaging over this number of simulation runs. The input layer consists of 3 nodes representing the state and the selected action to be taken. The output layer has one node. For each Monte-Carlo simulation run, the locations of secondary links (i.e., transmitting and receiving nodes) are generated randomly.

Performance is evaluated based on measuring, as a function of the number of SUs available in the network, the change in average MOS of the SUs achieved at the convergence point, and in average total number of iterations needed for algorithms 1, 2 and 3 to be converged. Note that the acceptable MOS level in terms of end-user quality perception is 3.

The performance of four different systems was compared during simulations while all system performing a physical-layer CR adaptation: a system called "*Q-Learning-Individual Learning*" where all SUs implement standard Q-learning (perform algorithm 1) and new joined SU initializes its

TABLE 3.4: Simulation parameters

Video model Parameters [58]	
Frame inter-arrival time	Deterministic (100, 50 ms)
Frame rate	10, 20 fps
End-to-end delay constraint	150, 75 ms
Packets per frame	8
Packet size	Truncated Pareto distribution ($a = 1.2$, $k = 2.5$ ms)
Packet inter-arrival time	Truncated Pareto distribution ($a = 1.2$, $k = 53$ bytes)
FTP model Parameters [3]	
File size	Truncated Log-normal distribution ($\mu = 2$ and $\sigma = 0.722$ Mbytes)
File inter-reading time	Exponential distribution ($\lambda = 180$ seconds)
algorithm. 2 Parameters	
Learning rate (α)	0.01
Discounting factor (γ)	0.8
Mini-batch size	10
Replay memory size	100
Buffer size L	100
P	8
Neural Network Parameters	
Input layer nodes	4 (representing present state and selected action)
Number of Hidden layers	2 layers with 4 and 2 nodes each
Output layer node	1
Network Parameters	
PN target SINR	1 dB
PU transmit power	23 dBm
Noise power	-90 dB

Q-table from zero, the second system, called "*Q-Learning-Transfer learning*", where all SUs implement standard Q-learning (algorithm 1) and new joined SU exploits the transfer learning (algorithm 3) and initializes its Q-table using the Q-table entries of its nearest SU, the third system, called "*DQL-Individual Learning*", implements DQL (algorithm 2) for all SUs and initiates action-value function parameters of the joined SU randomly, and the last one called "*DQL-Transfer learning*" implements DQL for all SUs (algorithm 2) and initiates action-value function parameters of the joined

SU using the parameters of the nearest SU, which can be either video or data traffic user. While the first and the third systems perform individual learning for the joined SU, disregarding the intelligence acquired by SUs already in the network, the second and the last one adopt transfer learning approach implemented in algorithm 3.

Fig. 3.6 shows the average MOS of secondary network at convergence point, as a function of total number of SUs in the SN. It can be seen how the average network MOS decreases for all considered systems as the number of SUs increases. The reason for this is that to meet the interference constraints as the number of users increases, each SU tends to converge to a smaller SINR value and corresponding smaller MOS. It can be also seen that the average MOS is almost the same for all considered systems (with little negligible difference stemming from different random replacement of users in the network for each considered system). In particular, the figure shows that transfer learning enables accurate transferring of the representation of the environment to the new joined user, such that it achieves the same performance as the individual learning. Moreover, this figure shows that Q-learning and DQL converge to almost the same solution for the optimization problem developed in (3.13). The real difference between these two reinforcement algorithms is represented by the next figure, i.e. Fig 3.7.

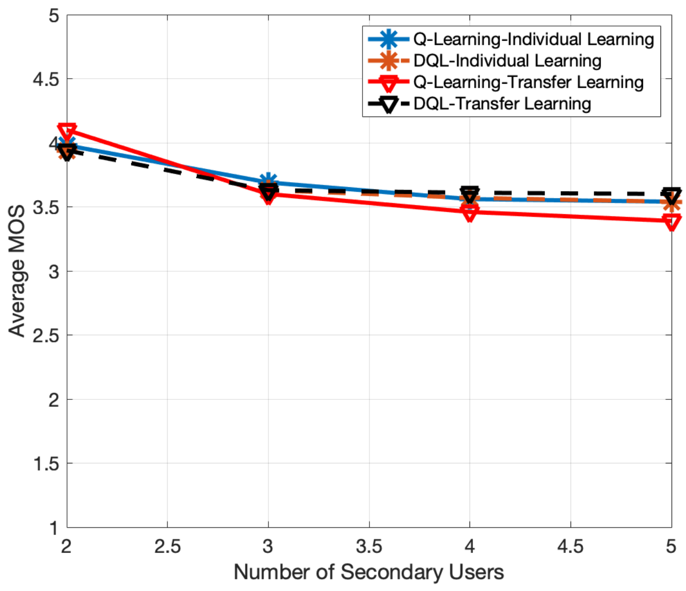

FIGURE 3.6: Average MOS.

Fig. 3.7 shows the efficiency of not only DQL but transfer learning compared to the Q-learning in accurately transforming the awareness of the surrounding environment to the new comer by experienced peers and reducing the number of iterations needed to achieve convergence. It can be seen that the number of iterations needed for Q-learning to achieve convergence utilizing transfer learning for the joined SU is reduced by as much as 75% compared to the individual learning. On the other hand, this figure shows the out-performance of DQL over Q-learning in converging faster to the optimal policy. In particular, the average number of iterations for the convergence increases for the Q-learning from 800 to 1600 while DQL needs iteration number in the range from 100 to around 1000. On the other hand, compared to the Q-learning, DQL leads to at most around 87% less iteration numbers to converge. According to this observations, for the cross-layer resource allocation design we just leverage the use of DQL and don't compare its performance with Q-learning. Instead, the results of cross-layer scenario are compared with single layer design to highlight the effect of cross-layer design in meeting the QoS constraint.

In the following, we consider our proposed cross-layer resource allocation technique designed for underlay cognitive radio networks and show its performance in terms of achieved average MOS for SN, its effect on the PN and specifically on the experienced QoS (i.e. end-to-end frame delay) for the

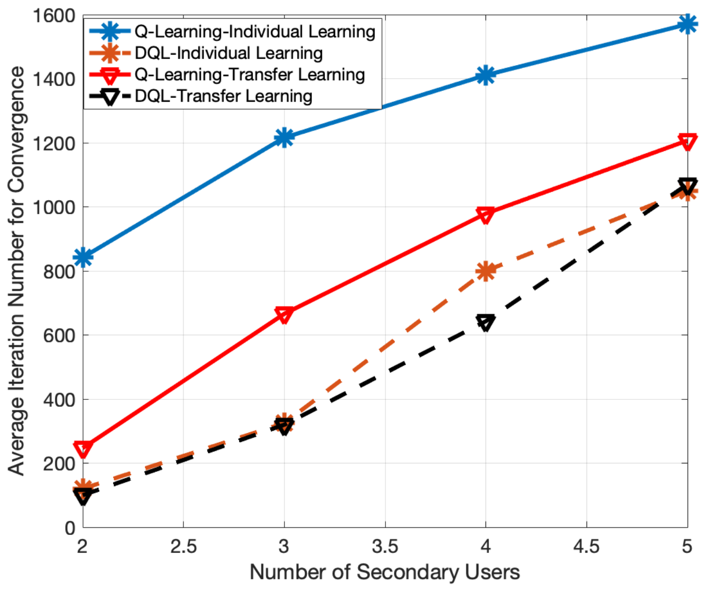

FIGURE 3.7: Average number of iteration for convergence.

video users. The secondary and primary network set up, distances between users, and their positioning policy is the same as the set up described for the single-layer technique. The main difference is that we model the video and data traffic to simulate a practical scenario, where we consider a buffer in our system design to queue the traffic packets and therefore be able to incorporate HARQ into our algorithm. Specifically, we examine the performance of DQL in solving the optimization problem developed in (3.18). The action set for each SU is $\mathcal{A} = \{-11, -7, -3, 1, 3, 5\}$ dB.

With regard to the learning algorithm the value of the learning rate α and discounting factor γ as in table 3.4 remain the same for all the SUs. In the ϵ-greedy exploration, as the number of iteration increases, ϵ converges to 0 from its initial value. The RL based resource allocation uses the feed forward neural network approach. Each SU uses two separate neural networks as action-value function and target action-value function approximators. Each of the feed forward network consists of two fully connected hidden layers that have 4 and 2 neurons each, input layer with 4 nodes and one node output layer. The four nodes in the input layer represent the present state and selected action.

The size of the replay memory and mini-batch has been listed in the table 3.4. For the simulation, it is assumed that each SU link carries only one type of traffic, based on the same percentage of random assignment of video streaming or FTP traffic type. The parameters of the FTP model [3] are the size of the transferred files and their inter-reading times which follow a truncated Log-normal distribution (with $\mu = 2$ Mbytes, $\sigma = 0.722$ Mbytes and maximum size of 5 Mbytes) and an exponential distribution (with $\lambda = 180$ seconds), respectively. The video streaming traffic model is from [58] where each video frame has a fixed number of packets. Each packet arrives at a random time interval while the inter-arrival time for each frame is deterministic. To model the size and the inter arrival time of packets in a frame, we use the truncated Pareto distribution. For the packet size, the truncated Pareto distribution parameters are set to $a = 1.2$ and $k = 53$ bytes while the mean, maximum and minimum packet size are 100, 250 and 53 bytes, respectively. The truncated Pareto distribution parameters for packet inter arrival time are set to $a = 1.2$ and $k = 2.5$ ms and the mean, maximum and minimum packet inter arrival time are

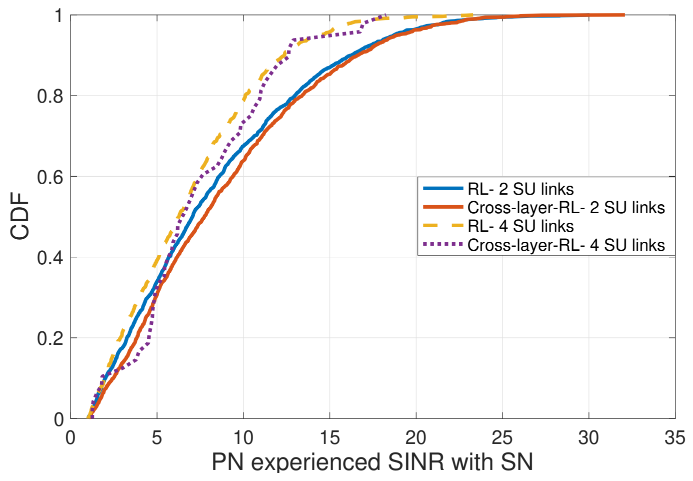

FIGURE 3.8: Experienced SINR at the PN after adding SN.

6, 12.5 and 2.5 ms, respectively. We consider two different video frame rates to investigate the adaptability of the resource allocation algorithm to the changes in delay requirements. The video frame rates are 10 (with 100 ms frame inter-arrival time) and 20 fps (with 50 ms frame inter-arrival time) and each with a delay constraint set to be one and half video frame inter-arrival time (75 ms and 150 ms for 20 and 10 fps, respectively). The size of the finite buffer (queue) L is set at 100 packets for all SUs, and P is assumed identical to the number of packets in a video frame ($P = 8$). To assess the error rate of the received transport block, and create the ACK/NACK feedback messages at SBSs to feed back to the SUs, a set of AWGN LTE link-level performance curves generated with [133] are used.

In order to investigate the effect of cross-layer design on the QoE (and particularly the delay) the performance of two different schemes is compared during simulations: the first scheme is the "*Cross-Layer RL*" where all the SUs perform the proposed cross-layer adaptation learning technique and the second scheme is the "*RL*" where all the SUs implement only physical-layer adaptation learning using DQL (without taking into account the frame delay as the additional element affecting the learning algorithm, i.e. w_1 always equal to 0 for all type of traffics).

Fig. 3.8 shows the CDF for the mean experienced SINR for the PN at the convergence point after the introduction of the SN. The QoE requirement for the SU should be met but not at the cost of disturbing PU transmission. It can be seen that both schemes meet the constraint on SINR for the PN (no less than 1 dB) but that the PN tends to achieve lower SINR as the number of SUs and consequently the interference to the PN increases. For instance, the probability of experiencing SINR at the PU less than 15 dB is 82% for SN with 2 SU links, while it is around 98% for SN with 4 SU links. It can be also seen that for the SN with 2 SU links the maximum achieved SINR at PU receiver is around 32 dB for cross-layer approach while it is around 28 dB for single-layer resource allocation. The case is vise versa for more congested SN, as for SN with 4 SU links single-layer approach achieves higher SINR up to 24 dB while it is around 18 dB for cross-layer approach. In general, both schemes meet the SINR constraint from PU link.

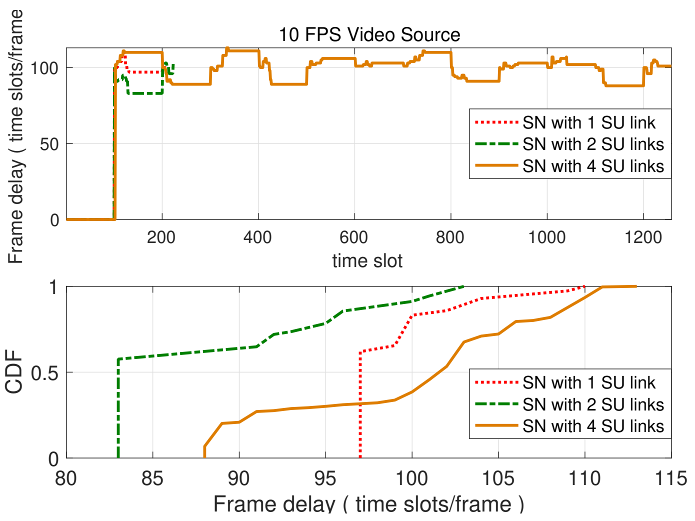

FIGURE 3.9: Frame delay for 10 fps video source.

Figs. 3.9 and 3.10 show the delay experienced delay at a randomly picked SU video over a period of eight preceding transmitted packets (denoted as frame delay in all figures) as a function of time slot, with each time slot being 1ms long. The results are depicted for one Monte-Carlo simulation run across different SN with 1, 2 and 4 SUs links. Fig. 3.9 shows that for 10 FPS video source (demanding an end-to-end frame delay less than 150 ms) the frame delay is always preserved less than 115 ms for all SNs. This figure's top part shows that as the number of SUs in SN increases the time needed for the algorithm to converge increases. In particular, the convergence time for SN with 1 SU is around 200 ms while it is around 1200 ms for SN with 4 SU links. This is because for higher number of SUs, the search space becomes bigger and it takes longer time for the RL agent

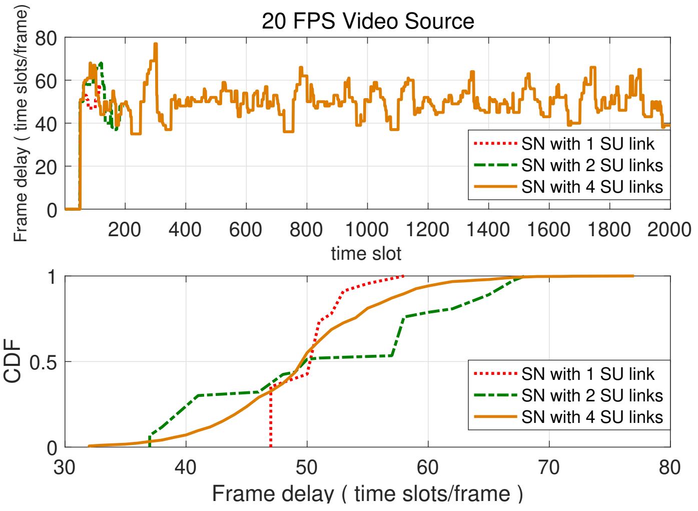

FIGURE 3.10: Frame delay for 20 FPS video source.

to find the best policy. Our condition for the algorithm to converge is that for three consecutive time slots not only the state remains the same and equal to (0,0,0) but the selected action remains unchanged. The bottom part of this figure shows the cumulative distribution function (CDF) for the frame delay. It can be clearly seen that for all SNs the frame delay is always less than 115 ms. Fig. 3.10 shows frame delay for 20 FPS video source (demanding an end-to-end frame delay less than 75 ms). The top part shows the variations in the frame delay as the network gets trained. The bottom part shows that for SN with 4 SU links the frame delay is less than 60 ms around 95% of the time. In general these two figures clearly show the effectiveness of our cross-layer algorithm in satisfying video traffic delay requirements.

Based on results averaged over 2000 Monte-Carlo simulation runs, Figs. 3.11 and 3.12 show the cumulative distribution function (CDF) for the average experienced frame delay over the video SUs. The simulation has a SN setup with 2 and 4 SU links. Fig. 3.11 depicts the mean frame delay for a 10 fps video source. For the first case, graph on top shows that for the SN with 2 SU links, both the learning techniques have the same performance and keep the frame delay less than 140 ms for more than 95% of the time. Thus in the case of a sparse SU distribution, which is hardly the case, both the techniques fare well. However, considering the practical scenario, where there will be a moderate to high number of SUs, the cross layer approach is the clear winner. The bottom graph indicates that as the network congestion increases and the number of SUs increases, the cross-layer approach performs better than the physical-layer technique. It can be seen that for the SN with 4 SUs, the cross-layer technique results in an experienced mean frame delay always less than the assigned 10 fps video source delay constraint (i.e. 150 ms). On the other hand, the mean frame delay increases up to around 550 ms for the physical-layer learning approach. The main reason for this is that the physical-layer learning technique does not take into account the experienced frame delay over the time in selecting the actions and updating the policy.

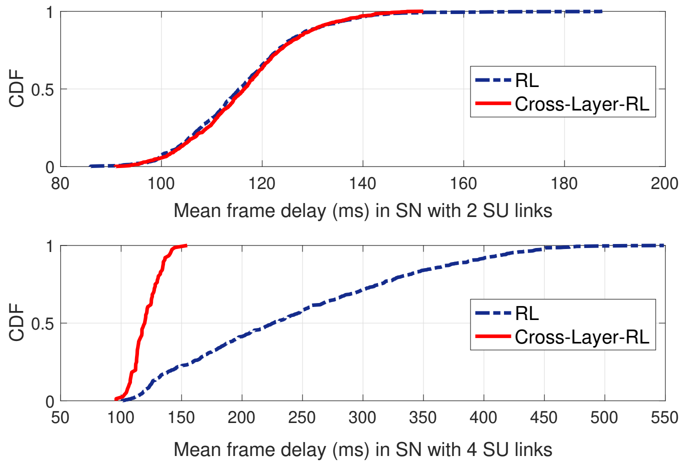

FIGURE 3.11: Mean frame delay for 10 fps video source.

For a 20 fps video source demanding a frame delay less than 75 ms, Fig 3.12 shows that a SN setup with 2 SU links with a cross-layer scheme, has an average frame delay maintained at less than 75 ms for around 98% of the time while it is 75% of the time for the physical-layer scheme. Thus, for a higher fps video source, but a fewer number of SU links in the network, the two techniques no longer perform at par, as in the first case of a 10 fps video source. As the number of SUs increases, Fig 3.12 at the bottom shows that the cross-layer learning technique acts even more aggressively and keeps the mean frame delay always less than the demanded delay constraint, while for the physical-layer technique the mean frame delay is less than 75 ms for only 20% of the time. In general these two figures clearly show the effectiveness of our proposed cross-layer learning technique in satisfying the video traffic delay requirements. In other words the figure demonstrates the

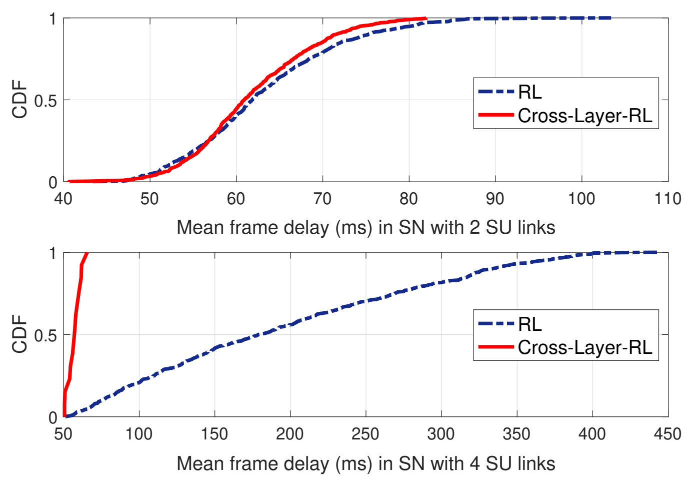

FIGURE 3.12: Mean frame delay for 20 fps video source.

effectiveness of our proposed technique in adaptation to the changes in the video source delay constraints and subsequently the QoE requirements.

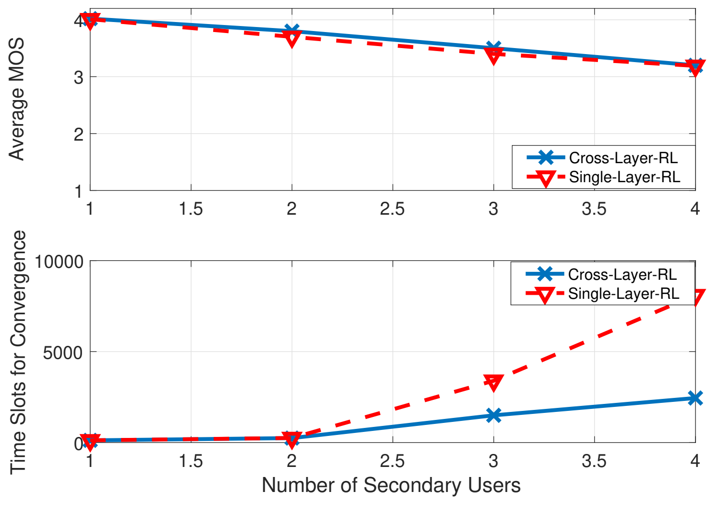

FIGURE 3.13: Average MOS and number of time slots at the convergence point in secondary network.

Fig. 3.13 depicts the performance of the algorithm in terms of the average MOS at the top, and of the average total number of iterations (time slots) needed for the convergence at the bottom, as a function of the number of SU links in the SN. It can be seen that for both learning techniques, the average network MOS decreases as the number of SUs increases. This is because each SU needs to operate at a smaller SINR value in order to meet the interference constraints with increasing network density, which results in a smaller MOS. It can be thereby deduced from the figure that the average MOS is maintained even for increased SUs in the network but with a slight deterioration. The figure at the bottom shows the number of iterations needed for the convergence of both schemes. It can be clearly deduced from the graph that the number of time slots required for the convergence point in the SN for a physical layer approach is way more than the cross-layer technique.

For a total of 4 SUs in the network, the RL approach takes around 1300 time slots in comparison to around 200 time slots for the cross-layer approach which is close to 6 times more. In other words, compared to the physical-layer approach, the cross-layer approach results in 80% reduction in the number of iterations for convergence, without affecting the average MOS performance. Due to the higher number of constraints for the cross-layer approach to satisfy, the exploration phase for this technique results in higher number of actions violating the constraints (i.e. $I_{t+1}+L_{t+1}+O^i_{t+1} > 0$) and consequently achieving negative total rewards $r^i_T < 0$. Since negative total reward for each action results in the lower action value function output, these actions will then be dropped in the exploitation phase of the learning process (according to the action selection strategy in algorithm 2 where the action with maximum action-value function output is selected). The cross-layer learning approach leads to lesser number of iterations compared to the physical-layer technique to find the global maximum. This is because of the fact that the cross-layer approach explores lesser number of remaining actions after the exploration phase.

In the following, we examine the effect of transfer learning on the performance of cross-layer underlay resource allocation. We evaluate the performance from the results of over 2000 Monte-Carlo simulation runs based on measuring, as a function of the number of SN links, the average MOS achieved at the convergence point, and the average total number of iterations needed for convergence of the adopted learning algorithm (i.e. DQL).

The performance of three different systems was compared during simulations with all performing the cross-layer queue-aware transmit parameter adaptation technique. All systems depict a scenario in which a new SU, that we call a "newcomer", becomes actives and adds a link to the SN where there are already other links active with SUs that have already gone through a converged learning process. We consider those SUs that were

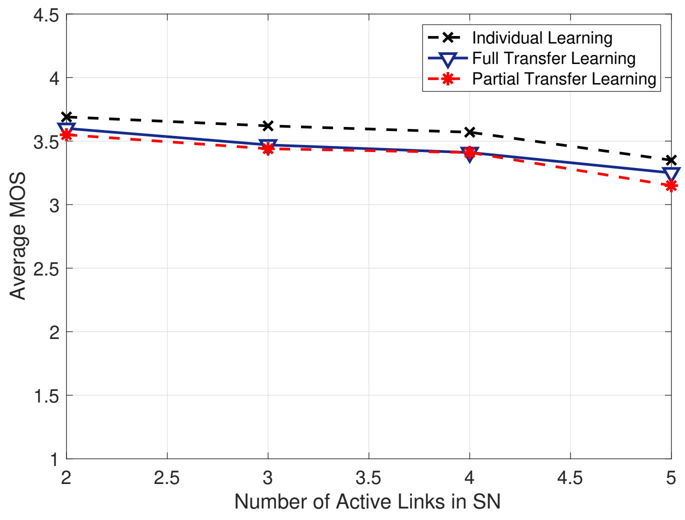

FIGURE 3.14: Average MOS in the secondary network.

already present as "experienced" and the newcomer as a naive one that needs to learn the resource allocation policy. The three systems that are considered investigate the effects of both partial and full transfer of knowledge, as compared to the no transfer of knowledge. The first system called "*Individual Learning*" represents the case where a newcomer SU initiates the weights of its action-value function neural networks randomly and disregards the experience acquired by the SUs already active in the network. The second system, called "*Full Transfer Learning*", corresponds to the case when the newcomer SU initializes the weights of its action-value function neural network using the entire parameters of the first nearest SU (ignoring the type of traffic that is carrying). The third and last system, called "*Partial Transfer Learning*", represents the case when the newcomer initializes the weights of the first two layers of its action-value function neural network with random values, while the last layer is initialized with the weights of the nearest SU's action-value function neural network.

While the first system performs individual learning for the joined SU and re-run algorithm 2, disregarding the intelligence acquired by SUs already in the network, the other two systems initializes the parameters of the action-value function of the newcomer SU through the transfer learning approach (i.e. algorithm 3).

Fig. 3.14 shows the average MOS of the SN after convergence of the learning algorithm, as a function of the number of SU links in SN. The results show

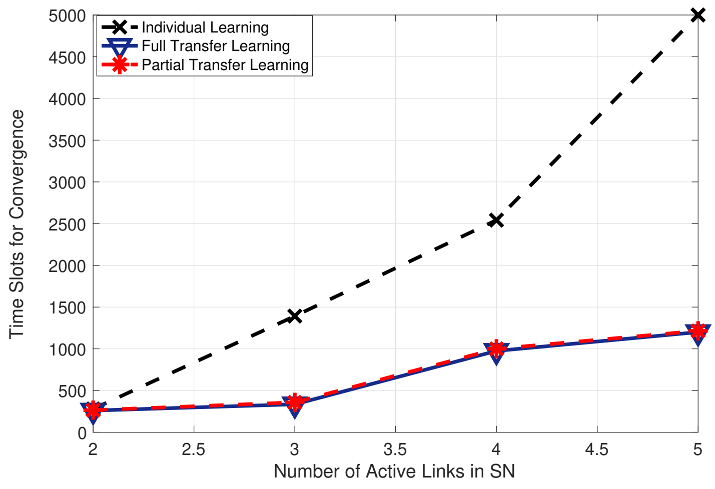

FIGURE 3.15: Average number of iterations needed for algorithm convergence.

that there is practically no negative effect on the achieved QoE when using transfer learning, as the MOS of the transfer learning system is within a difference of 0.1 MOS from the system with individual learning (and not transfer learning). Moreover, this result shows no difference in MOS between full and partial transfer learning.

Fig. 3.15 shows, as the function of the number of active links in the SN, the number of iterations needed for the DQL algorithm to converge. It clearly shows that compared to the individual learning the transfer learning needs a much smaller number of iterations to converge. It can be also seen that as the number of SUs increases the relative reduction in the number of iterations for transfer learning with respect to the individual learning case increases. For example for the SN with 4 active links, the number of iterations needed to achieve convergence is reduced by as much as 80% when deploying transfer learning (partial as well as full) compared to the case where the newcomer SU performs individual learning. More importantly, as noted in Fig. 3.14, a partial transfer learning has essentially the same MOS performance as the full transfer learning scheme. Consequently, the scheme with partial transfer learning which exchanges only the weights of the third layer of the expert SU's action-value function neural network reduces the communication overhead by around 90%.

Together, Figs. 3.14 and 3.15 show the effectiveness of the partial transfer learning scheme in accurately carrying the awareness of the surrounding environment from an expert peer SU to the newcomer CR, leading to significantly fewer number of iterations needed to achieve convergence, while also reducing the communication overhead by a large magnitude. More importantly, the significant communication overhead savings that is achieved by exchanging the parameters of only the third layer of the action-value function neural network, not only exhibits the same average MOS performance as the full transfer learning scheme, but it also shows no practical difference in MOS performance with respect to the case of individual learning by the newcomer SU (no exchange of knowledge).

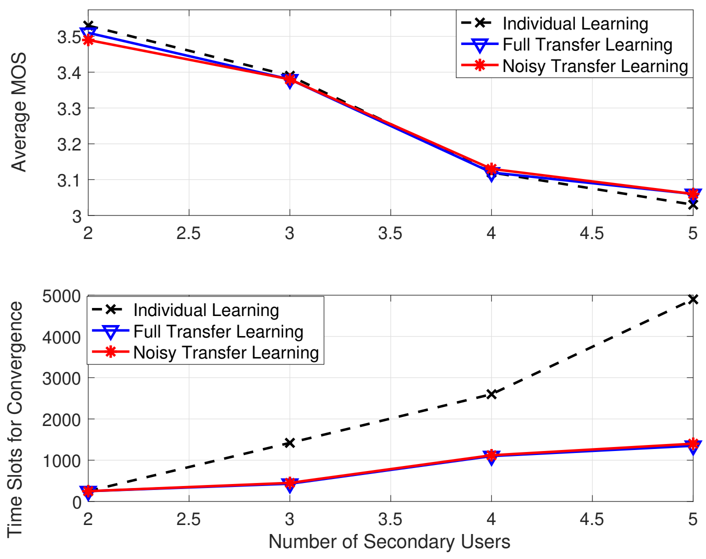

Figure 3.16: Average MOS and average number of iterations needed for algorithm convergence.

Moreover, we explored a practical scenario where the noisy version of the weights are transferred to the new joined SU (which it can be also seen as imperfect transferring of the weights). Such noisy weights can be created through the wireless channel during transferring to the new SU, and/or during quantization process at the transmitter of the node from which the weights are transferred. We only considered the scenario where the quentization noise is added to the weights. To implement this scenario, we considered a simple approach where the weights greater than zero are mapped to 0.5 and the remaining are mapped to −0.5. Specifically, only one bit is used to transfer each weight, and consequently binary modulation scheme (such as BPSK) can be used to modulate the sequence of weights. To explore the effect of imperfect transferring of the weights and compare its performance with perfect weight transferring, performance of three different systems was compared during simulations, with all performing the cross-layer resource allocation technique discussed in this chapter. The three systems that are considered investigate the effects of both noisy (imperfect) and perfect transfer of knowledge, as compared to the no transfer of knowledge. The first system called "*Individual Learning*" represents the case where a newcomer SU initiates the weights of its action-value function neural networks randomly and disregards the experience acquired by the SUs already active in the network. The second system, called "*Full Transfer Learning*", corresponds to the case when the newcomer SU initializes the weights of its action-value function neural network using the noiseless parameters of the first nearest SU. The third system, called "*Noisy Transfer learning*", represents the case when the newcomer SU initializes the weights of its action-value function neural network using the noisy (quantized) version of its first nearest SU's parameters. In fact, the first system disregards the intelligence achieved by other users already in the system and rerun the cognition cycle from the scratch, while the two other systems take advantage of transfer learning and initialize the parameters of the action-value function of the newcomer SU through the (full and noisy) transfer learning approach.

Fig. 3.16 depicts as the function of the number of SU links the performance of the three considered algorithm in terms of the average MOS at the top, and of the average total number of iterations (time slots) needed for the convergence at the bottom. It can be seen that for all systems, the average network MOS is almost the same. On the other hand, neither full transfer learning nor noisy transfer learning compromise the MOS and result in the same MOS as individual learning. It can be also seen that the noisy transferring of the weights results in the same number of iterations for the convergence as the full transfer learning. To make sure about the observed results regarding the number of iterations for the convergence, we monitored the changes in Q-values at each state across every action, and we noticed that the slop of the change in Q-values is the same between full transfer and noisy transfer learning schemes, which results in the same speed in convergence to the optimal policy at each state for both schemes.

Fig. 3.17 shows for one randomly selected SU the changes in the weight

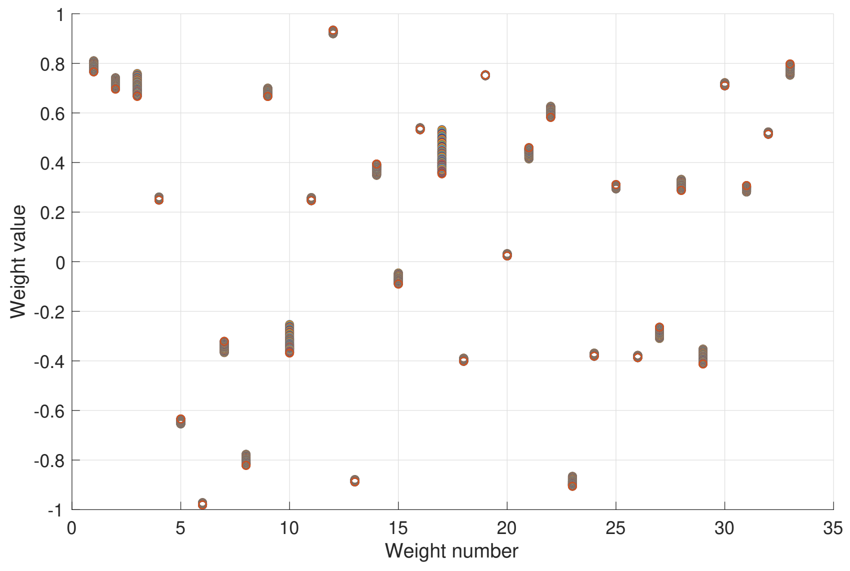

FIGURE 3.17: Average MOS and average number of iterations needed for algorithm convergence.

value for all 33 weights in action-value neural network estimating action-value function, during the training process. For each weight in this figure, the blue circle and red circle show the initial random weight value and convergence point weight value (value for each weight at the convergence point), respectively. In particular, this figure depicts for one Monte-Carlo simulation run the value to which each weight is changed, at each 10 training rounds. The figure shows the disparity of the weight values between a span of 1 to -1. To implement the full transfer learning (i.e. transferring the exact amount for weight values) for just this snapshot of the network, at least 6 bits are needed. While with adoption of noisy transfer learning, only one bit is needed for each weight to be transferred across every snapshot of the network. In general, noisy transferring of the weights results in not only preserving the average QoE the same as the full transferring scenario, but major reduction in communication overhead.

3.8 Summary

In accordance with the evolution of resource management techniques for 5G networks, in this chapter we presented two underlay DSA techniques that allow to adapt transmit power, modulation scheme and accordingly transmit rate of all SUs to maximize average QoE across all traffics of dissimilar characteristics (real-time video and regular data traffic) in the secondary network while satisfying the interference constraint to the PU transmission. The first proposed resource allocation technique maps the QoS parameters of the data traffic (which is transmit rate) and video traffic (which is PSNR) to the QoE parameters and is designed to maximize the average experienced QoE in the SN while meeting the underlay DSA constraint. The second technique takes into account an additional QoS parameter for the video traffic which is end-to-end video frame delay. Specifically, in addition to underlay DSA constraint, the second proposed technique applies a new constraint on the experienced end-to-end video frame delay and manages to satisfy both constraints through a cross-layer approach, while maximizing the average QoE across all video and data transmissions in the SN. MOS is exploited as an metric to model the subjective QoE as it not only meets the end-user centric quality assessment requirements, but enables the seamless integration of dissimilar traffic by providing a single common measuring scale for different types of traffic.

a reinforcement learning algorithm is selected to find the solution to the developed optimization problems in this chapter. In particular, Q-learning and DQL are adopted to learn the underlying dynamics model of the considered systems in this chapter. Simulation results show that DQL outperforms Q-learning in terms of the convergence speed. Specifically, simulation results demonstrated that the DQL needs around 87% less iterations to converge to the optimal solution, without any compromise in MOS. This observation led to explore only the deployment of DQL-based learning framework for the next proposed cross-layer algorithm. The cross-layer algorithm adapted transmission power and rate of the SUs to their end-to-end traffic delay requirements and to the underlay DSA interference constraint. In doing so, the algorithm was designed to maximize the overall MOS in the SN across all data and video transmissions. To validate the effectiveness of the proposed cross-layer resource allocation algorithm, we deployed a single-layer resource allocation as well and conducted extensive simulations to monitor the frame delay and QoE in SN.

Simulation results show that under practical constraints of finite buffer size and variable packet length the proposed cross-layer technique satisfies the interference constraints for the PN which is the main consideration in the underlay DSA. Moreover, the proposed technique effectively adapts to the dynamics of the video traffic delay with significantly (around 80%) fewer number of iterations required for the convergence in comparison to the single-layer technique. Having achieved convergence with fewer iterations, it is of notable significance that the average MOS performance is not affected in the process.

In addition, to improve the convergence time of the DQL algorithm we applied the idea of transfer learning which allows newcomer SUs learn from their more expert peers to improve the learning process. The philosophy behind transfer learning is to allow well-established and expert cognitive agents (i.e. base stations or mobile stations in context of wireless communications) to teach newly activated and naive agents. In the proposed transfer learning schemes, when a newcomer SU node joins the SN, it initializes its DQN-based cognitive engine from the representation of the wireless environment (in the form of transferring the weights of its action-value function neural network) that has already been learned by the node that was already active in the network. This exchange of learned information is used to improve the performance of a distributed cognitive radio networks. We further identified the best practices to transfer learning between cognitive radios so as to reduce communication overhead and to identify the node from which to transfer knowledge. To decide about the node from which to transfer knowledge from, we examined the similarity (in terms of mean square error) between the action-value function parameters of SUs in an already trained SN implementing the resource allocation. We observed that as the distance between SUs increases, the similarity between the parameters decreases. Therefore, for the joined SU the nearest SU (the one which is received with strongest signal and might be either video or data user) is the best expert user to learn from. In order to reduce communication overhead, we explored the effect of partial transfer of the nearest SU's action-value function parameters to the new joined one.

Simulation results show that both full and partial transfer learning reduces the number of iterations for convergence by approximately 80% without a sacrifice in average QoE performance. Importantly, the results demonstrate the fact that partial transfer learning, in which parameters of only the third layer of the expert user's action-value function is exchanged with the newcomer SU, leads to a significant reduction (around 90%) in communication overhead, without compromising the average MOS in the SN.

We also studied the effect of transferring the imperfect (noisy) version of the weights to the joined SU, through defining one quantization level and quantizing the weights and then transferring them through 1 bit of information. The simulation results demonstrated that this process not only preserved the average QoE the same as the perfect transferring of the weights, but also reduced at a great extend the communication overhead.

Chapter 4

Conclusion

Cognitive radio (CR) has been used to refer to wireless communication devices that are capable of autonomously learning, reasoning and adapting to their environment. These capabilities are embedded in a cognitive engine which has been identified as the core of a CR. The cognitive engine adapts the actions (parameters) of the CR. This task become significantly challenging when several parameters and policies need to be adjusted simultaneously (e.g. transmit power, transmit rate, coding scheme, modulation scheme, sensing algorithm, sensing policy, etc.) and no simple formula may be able to simultaneously determine these setup parameters. This is due to the complex interactions among these factors and their impact on the wireless environment. Thus, learning methods can be applied to allow efficient adaption of the CRs to their environment, without the complete knowledge of the dependence among these parameters by leveraging the use of machine learning algorithms.

Ever increasing wireless application demands with different QoE requirements along with limited available spectrum led to the shift from static radio spectrum access to DSA and later to CR as DSA enabling technology. Among DSA techniques, the underlay DSA enables simultaneous coexistence of the cognitive radios / secondary users with the primary users on the same frequency band, subject to the limitation of the interference caused by the SUs to the primary links. Due to this limitation, effective resource allocation becomes more challenging for underlay cognitive radio networks, especially when the bandwidth greedy multimedia services that need provision of higher QoE are transmitted in the network. This problem of meeting two contradicting goals of maximizing the QoE and satisfying the interference constraint to the primary user can be modeled through discrete time Markov decision process (DTMDP). In case of unknown wireless environment (i.e. the transition probabilities of the Markov model are unknown), by applying special learning algorithms such as reinforcement learning, it is possible to arrive at the optimal solution to the MDP [23].

In this thesis, we have proposed machine learning-enabled resource allocation techniques for underlay cognitive radio networks. Specifically, we leveraged the use of supervised learning and reinforcement learning to learn underlying model of the underlay wireless communication systems.

In chapter 1, we reviewed application of machine learning to the cognitive radio network. Next, we explained the three distinguished categories of the machine learning algorithms and highlighted the ones utilized in our proposed resource allocation algorithms. Lastly, we reviewed the existing learning efforts made to address issues in cognitive radio networks.

In chapter 2, we presented a fully autonomous and distributed underlay DSA technique that was based on a NARX neural network cognitive engine. The NARX neural network engine leveraged the use of adaptive modulation and coding in the primary network to infer the effect that different settings in a secondary network transmission would have on the primary network. In effect, the NARX neural network learned to use the sensed modulation used in the nearest primary link to predict the throughput (equivalently the chosen modulation scheme and the channel coding rate) resulting from a secondary link transmission setting in the same primary link. Based on this NARX neural network capability, we presented two variants for the proposed underlay dynamic spectrum access mechanism: one where the secondary users choose the maximum transmit power value that is estimated to not lead to a change in modulation order at their respective nearest primary link, and another where the secondary users choose the maximum transmit power value that is estimated to not change their respective nearest primary link throughput beyond a chosen maximum relative change value. The performance of the latter proposed underlay dynamic spectrum access mechanism was examined for the cases of sending all or a third of all probe messages to infer the effect of SU transmission on the transmission of the nearest PU. Simulation results showed that the proposed technique is able to accurately predict the modulation scheme and channel coding rate used in a primary link without the need to exchange information between the primary network and the secondary (e.g. access to feedback channels), resulting in the proposed technique succeeding in its main goal of determining the transmit power of the secondary users such that their created interference remained below the maximum threshold that the primary network could sustain with minimal effect on the average throughput, along with reducing the transmission overhead in case of sending fraction of probe messages. Through simulation results, it was shown how the proposed underlay DSA is able to control its effect on the primary network with a very fine level of granularity. At the same time, it was seen that our proposed algorithm was able to find transmit settings for the secondary users that will result in as large throughput in the secondary network as could be allowed by the primary network interference limit. Specifically, for a target PN maximum relative average throughput change of 2% the proposed scheme is able to maintain the PN relative throughput change less than 3% when sending all probe messages, and also less than 3.5% when reducing three times the number of transmitted probe messages, while at the same time achieving useful average throughput values in the secondary network between 180 and 50 kbps for a channel with 180 kHz bandwidth. We also discussed how the ability of our proposed technique to predict the full AMC mode (not just the modulation scheme) results in a significant increase in the transmission opportunities in the SN compared to schemes that only use the modulation classification information. Moreover, it was discussed how our proposed scheme based on a target primary network maximum relative average throughput change allows to manage the tradeoff between effect of the secondary network on the primary network and achievable throughput at the secondary network. Finally, we also discussed how the ability of our proposed technique to predict the full AMC mode (not just the modulation scheme) results in a significant increase in the transmission opportunities in the secondary network compared to schemes that only use the modulation classification information.

In chapter 3, we first presented a single-layer resource allocation technique for SN in heterogeneous underlay cognitive radio networks in order to maximize the average QoE for SN while meeting the PU SINR threshold. We leveraged the use of Q-learning as well as DQL to solve our optimization problem. Simulation results showed outperformance of DQL in comparison with Q-learning in terms of the iteration needed for the algorithm to converge. Next, we proposed a cross-layer resource allocation algorithm for underlay DSA network, where we only utilized the DQN-based learning framework to adapt transmission power and rate of the SUs to not only the underlay DSA interference constraint, but also end-to-end traffic delay requirements. The optimization problem was designed to maximize the overall QoE in the SN across all data and video transmissions. To validate the effectiveness of the proposed cross-layer resource allocation algorithm we deployed a single-layer resource allocation as well and conducted extensive simulations to monitor the frame delay and QoE in SN. We exploited the MOS to model the subjective QoE. The MoS model enables the seamless integration of dissimilar traffic assumed in this paper by providing a single common measuring scale for different types of traffic. We showed the outperformance of our proposed algorithm over the single-layer one. Moreover, we incorporated in our algorithm transfer learning to improve the convergence time, which allows a SU joining the network to initialize its action-value function parameters from their more experienced neighbors to accelerate the learning process. After completion of an initial learning exploration, the nearest node to the joined SU, which is already in the network, transfers its learned representation of the environment by transferring its action-value function parameters to the joined SU. Simulation results showed that under practical constraints of finite buffer size and variable packet length the proposed cross-layer technique satisfied the interference constraints for the PN which is the main consideration in the underlay DSA. Moreover, the proposed technique effectively adapted to the dynamics of the video traffic delay with significantly (around 80%) fewer number of iterations required for the convergence in comparison to the single-layer technique. Having achieved convergence with fewer iterations, the average MOS performanace was not affected in the process.

Bibliography

[1] Andrea Goldsmith, Syed Ali Jafar, Ivana Maric, and Sudhir Srinivasa. Breaking spectrum gridlock with cognitive radios: An information theoretic perspective. *Proceedings of the IEEE*, 97(5):894–914, 2009.

[2] Kandaraj Piamrat, Cesar Viho, Jean-Marie Bonnin, and Adlen Ksentini. Quality of experience measurements for video streaming over wireless networks. In *2009 Sixth International Conference on Information Technology: New Generations*, pages 1184–1189. IEEE, 2009.

[3] Farooq Khan. *LTE for 4G mobile broadband: air interface technologies and performance*. Cambridge university press, 2009.

[4] 3GPP. Feasibility study for orthogonal frequency division multiplexing (ofdm) for utran enhancement (release 6). TR25.892, version 6.0.0, June 2004.

[5] Marco Chiani, Andrea Giorgetti, and Gianluigi Liva. Ultra wide bandwidth communications towards cognitive radio. In *EMC Europe Workshop*, pages 114–117, 2005.

[6] Danijela Cabric and Robert W Brodersen. Physical layer design issues unique to cognitive radio systems. In *2005 IEEE 16th International Symposium on Personal, Indoor and Mobile Radio Communications*, volume 2, pages 759–763. IEEE, 2005.

[7] Matteo Gandetto and Carlo Regazzoni. Spectrum sensing: A distributed approach for cognitive terminals. *IEEE Journal on selected areas in communications*, 25(3):546–557, 2007.

[8] Allen Petrin and Paul G Steffes. Analysis and comparison of spectrum measurements performed in urban and rural areas to determine the

total amount of spectrum usage. In *Proc. International Symposium on Advanced Radio Technologies (ISART 2005)*, pages 9–12, 2005.

[9] Chunsheng Xin, Min Song, Liangping Ma, George Hsieh, and Chien-Chung Shen. Network coding relayed dynamic spectrum access. In *Proceedings of the 2010 ACM workshop on Cognitive radio networks*, pages 31–36, 2010.

[10] Zeljko Tabakovic, Sonja Grgic, and Mislav Grgic. Dynamic spectrum access in cognitive radio. In *2009 International Symposium ELMAR*, pages 245–248. IEEE, 2009.

[11] Joseph Mitola and Gerald Q Maguire. Cognitive radio: making software radios more personal. *IEEE personal communications*, 6(4): 13–18, 1999.

[12] Min Chen, Shiwen Mao, and Yunhao Liu. Big data: A survey. *Mobile networks and applications*, 19(2):171–209, 2014.

[13] Paulo Valente Klaine, Muhammad Ali Imran, Oluwakayode Onireti, and Richard Demo Souza. A survey of machine learning techniques applied to self-organizing cellular networks. *IEEE Communications Surveys & Tutorials*, 19(4):2392–2431, 2017.

[14] Sotiris B Kotsiantis, I Zaharakis, and P Pintelas. Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, 160:3–24, 2007.

[15] Paulo Valente Klaine, Oluwakayode Onireti, Richard Demo Souza, and Muhammad Ali Imran. The role and applications of machine learning in future self-organizing cellular networks. In *Next-Generation Wireless Networks Meet Advanced Machine Learning Applications*, pages 1–23. IGI Global, 2019.

[16] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[17] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker,

Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354–359, 2017.

[18] Mohssen Mohammed, Muhammad Badruddin Khan, and Eihab Bashier Mohammed Bashier. *Machine learning: algorithms and applications*. Crc Press, 2016.

[19] Stephen Marsland. *Machine learning: an algorithmic perspective*. Chapman and Hall/CRC, 2014.

[20] Liang Yin, SiXing Yin, Weijun Hong, and ShuFang Li. Spectrum behavior learning in cognitive radio based on artificial neural network. In *2011-MILCOM 2011 Military Communications Conference*, pages 25–30. IEEE, 2011.

[21] Achim Engelhart, Werner G Teich, Jürgen Lindner, Gábor Jeney, Sándor Imre, and László Pap. A survey of multiuser/multisubchannel detection schemes based on recurrent neural networks. *Wireless Communications and Mobile Computing*, 2(3):269–284, 2002.

[22] Mohammed Alenezi, Kok Keong Chai, Shihab Jimaa, and Yue Chen. Use of unsupervised learning clustering algorithm to reduce collisions and delay within lora system for dense applications. In *2019 International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 1–5. IEEE, 2019.

[23] Richard S Sutton and Andrew G Barto. Reinforcement learning: an introduction cambridge. *MA: MIT Press.*, 1998.

[24] Yonghua Wang, Zifeng Ye, Pin Wan, and Jiajun Zhao. A survey of dynamic spectrum allocation based on reinforcement learning algorithms in cognitive radio networks. *Artificial Intelligence Review*, 51 (3):493–506, 2019.

[25] Peter Dayan and CJCH Watkins. Q-learning. *Machine learning*, 8 (3):279–292, 1992.

[26] Richard Bellman. The theory of dynamic programming. Technical report, Rand corp santa monica ca, 1954.

[27] Pedro Amaral, Joao Dinis, Paulo Pinto, Luis Bernardo, Joao Tavares, and Henrique S Mamede. Machine learning in software defined networks: Data collection and traffic classification. In *2016 IEEE 24th International Conference on Network Protocols (ICNP)*, pages 1–5. IEEE, 2016.

[28] Dario Rossi and Silvio Valenti. Fine-grained traffic classification with netflow data. In *Proceedings of the 6th international wireless communications and mobile computing conference*, pages 479–483, 2010.

[29] Akihiro Nakao and Ping Du. Toward in-network deep machine learning for identifying mobile applications and enabling application specific network slicing. *IEICE Transactions on Communications*, 2018.

[30] Mostafa Uddin and Tamer Nadeem. Trafficvision: A case for pushing software defined networks to wireless edges. In *2016 IEEE 13th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, pages 37–46. IEEE, 2016.

[31] Pu Wang, Shih-Chun Lin, and Min Luo. A framework for qos-aware traffic classification using semi-supervised machine learning in sdns. In *2016 IEEE International Conference on Services Computing (SCC)*, pages 760–765. IEEE, 2016.

[32] Rejab Hajlaoui, Hervé Guyennet, and Tarek Moulahi. A survey on heuristic-based routing methods in vehicular ad-hoc network: Technical challenges and future trends. *IEEE Sensors Journal*, 16(17): 6782–6792, 2016.

[33] Sandra Sendra, Albert Rego, Jaime Lloret, Jose Miguel Jimenez, and Oscar Romero. Including artificial intelligence in a routing protocol using software defined networks. In *2017 IEEE International Conference on Communications Workshops (ICC Workshops)*, pages 670–674. IEEE, 2017.

[34] Giorgio Stampa, Marta Arias, David Sánchez-Charles, Victor Muntés-Mulero, and Albert Cabellos. A deep-reinforcement learning approach for software-defined networking routing optimization. *arXiv preprint arXiv:1709.07080*, 2017.

[35] Frederic Francois and Erol Gelenbe. Optimizing secure sdn-enabled inter-data centre overlay networks through cognitive routing. In *2016 IEEE 24th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, pages 283–288. IEEE, 2016.

[36] Abdelhadi Azzouni, Raouf Boutaba, and Guy Pujolle. Neuroute: Predictive dynamic routing for software-defined networks. In *2017 13th International Conference on Network and Service Management (CNSM)*, pages 1–6. IEEE, 2017.

[37] Josep Carner, Albert Mestres, Eduard Alarcón, and Albert Cabellos. Machine learning-based network modeling: An artificial neural network model vs a theoretical inspired model. In *2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN)*, pages 522–524. IEEE, 2017.

[38] Rafael Pasquini and Rolf Stadler. Learning end-to-end application qos from openflow switch statistics. In *2017 IEEE Conference on Network Softwarization (NetSoft)*, pages 1–9. IEEE, 2017.

[39] Yanjiao Chen, Kaishun Wu, and Qian Zhang. From qos to qoe: A tutorial on video quality assessment. *IEEE Communications Surveys & Tutorials*, 17(2):1126–1165, 2014.

[40] Tasnim Abar, Asma Ben Letaifa, and Sadok El Asmi. Machine learning based qoe prediction in sdn networks. In *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 1395–1400. IEEE, 2017.

[41] Mingzhe Chen, Mohammad Mozaffari, Walid Saad, Changchuan Yin, Mérouane Debbah, and Choong Seon Hong. Caching in the sky: Proactive deployment of cache-enabled unmanned aerial vehicles for optimized quality-of-experience. *IEEE Journal on Selected Areas in Communications*, 35(5):1046–1061, 2017.

[42] Chenmeng Wang, Ying He, F Richard Yu, Qianbin Chen, and Lun Tang. Integration of networking, caching, and computing in wireless systems: A survey, some research issues, and challenges. *IEEE Communications Surveys & Tutorials*, 20(1):7–38, 2017.

[43] Ru Huo, Fei Richard Yu, Tao Huang, Renchao Xie, Jiang Liu, Victor CM Leung, and Yunjie Liu. Software defined networking, caching, and computing for green wireless networks. *IEEE Communications Magazine*, 54(11):185–193, 2016.

[44] Ying He, F Richard Yu, Nan Zhao, Victor CM Leung, and Hongxi Yin. Software-defined networks with mobile edge computing and caching for smart cities: A big data deep reinforcement learning approach. *IEEE Communications Magazine*, 55(12):31–37, 2017.

[45] Yuehong Gao, Lei Cheng, Lin Sang, Dacheng Yang, et al. Spectrum sharing for lte and wifi coexistence using decision tree and game theory. In *2016 IEEE Wireless Communications and Networking Conference*, pages 1–6. IEEE, 2016.

[46] Nasrin Sultana, Naveen Chilamkurti, Wei Peng, and Rabei Alhadad. Survey on sdn based network intrusion detection system using machine learning approaches. *Peer-to-Peer Networking and Applications*, 12(2):493–501, 2019.

[47] Nathan Shone, Tran Nguyen Ngoc, Vu Dinh Phai, and Qi Shi. A deep learning approach to network intrusion detection. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2(1):41–50, 2018.

[48] Ping Wang, Kuo-Ming Chao, Hsiao-Chung Lin, Wen-Hui Lin, and Chi-Chun Lo. An efficient flow control approach for sdn-based network threat detection and migration using support vector machine. In *2016 IEEE 13th International Conference on e-Business Engineering (ICEBE)*, pages 56–63. IEEE, 2016.

[49] Xin Kang, Rui Zhang, Ying-Chang Liang, and Hari Krishna Garg. Optimal power allocation strategies for fading cognitive radio channels with primary user outage constraint. *IEEE Journal on Selected Areas in Communications*, 29(2):374–383, 2011.

[50] Leila Musavian and Sonia Aïssa. Capacity and power allocation for spectrum-sharing communications in fading channels. *IEEE Transactions on Wireless Communications*, 8(1):148–156, 2009.

[51] A. Ghasemi and E. S. Sousa. Capacity of fading channels under spectrum-sharing constraints. In *2006 IEEE International Conference on Communications*, volume 10, pages 4373–4378, June 2006.

[52] Yuli Yang, Hao Ma, and Sonia Aissa. Cross-layer combining of adaptive modulation and truncated arq under cognitive radio resource requirements. *ieee transactions on vehicular technology*, 61(9):4020–4030, 2012.

[53] K. Eswaran, M. Gastpar, and K. Ramchandran. Bits through arqs: Spectrum sharing with a primary packet system. In *2007 IEEE International Symposium on Information Theory*, pages 2171–2175, June 2007.

[54] J. C. F. Li, W. Zhang, A. Nosratinia, and J. Yuan. Sharp: Spectrum harvesting with arq retransmission and probing in cognitive radio. *IEEE Transactions on Communications*, 61(3):951–960, March 2013.

[55] Zouheir Rezki and Mohamed-Slim Alouini. Ergodic capacity of cognitive radio under imperfect channel-state information. *IEEE Transactions on Vehicular Technology*, 61(5):2108–2119, 2012.

[56] E Kayalvizhi and Balamurugan Gopalakrishnan. Estimation of optimal channel gain in cognitive radio networks using bisectional algorithm. *International Journal of Advanced Networking and Applications*, 11(1):4171–4176, 2019.

[57] 3rd Generation Partnership Project Std. S25.214 V5.11.0. 3GPP Technical Specification Group Radio Access Network Physical layer procedures (FDD) (Release 5). Technical report, 2005.

[58] ANSI/IEEE Std. 802.11b 1999 (R2003). Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Higher-Speed Physical Layer Extension in the 2.4 GHz Band. Technical report, 1999.

[59] R. Zhang. On active learning and supervised transmission of spectrum sharing based cognitive radios by exploiting hidden primary radio feedback. *IEEE Transactions on Communications*, 58(10):2960–2970, October 2010.

[60] Lin Zhang, Ming Xiao, Gang Wu, Guodong Zhao, Ying-Chang Liang, and Shaoqian Li. Proactive cross-channel gain estimation for spectrum sharing in cognitive radio. *IEEE Journal on Selected Areas in Communications*, 34(10):2776–2790, 2016.

[61] Chadi Tarhini and Tijani Chahed. On capacity of ofdma-based ieee802. 16 wimax including adaptive modulation and coding (amc) and inter-cell interference. In *Local & Metropolitan Area Networks, 2007. LANMAN 2007. 15th IEEE Workshop on*, pages 139–144. IEEE, 2007.

[62] James Yang, Noel Tin, and Amir K Khandani. Adaptive modulation and coding in 3g wireless systems. In *56th Vehicular Technology Conference*, volume 1, pages 544–548. IEEE, 2002.

[63] Fuhui Zhou, Yongpeng Wu, Rose Qingyang Hu, Yuhao Wang, and Kai-Kit Wong. Energy-efficient noma enabled heterogeneous cloud radio access networks. *arXiv preprint arXiv:1801.01996*, 2018.

[64] C. M. Spooner and W. A. Gardner. The cumulant theory of cyclostationary time-series. ii. development and applications. *IEEE Transactions on Signal Processing*, 42(12):3409–3429, Dec 1994.

[65] A. Fehske, J. Gaeddert, and J. H. Reed. A new approach to signal classification using spectral correlation and neural networks. In *First IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005.*, pages 144–150, Nov 2005.

[66] H. Abuella and M. K. Ozdemir. Automatic modulation classification based on kernel density estimation. *Canadian Journal of Electrical and Computer Engineering*, 39(3):203–209, Summer 2016.

[67] Xiaoxin Qiu and K. Chawla. On the performance of adaptive modulation in cellular systems. *IEEE Transactions on Communications*, 47(6):884–895, Jun 1999.

[68] C. Mehlführer, M. Wrulich, J. C. Ikuno, D. Bosanska, and M. Rupp. Simulating the long term evolution physical layer. In *2009 17th European Signal Processing Conference*, pages 1471–1478, Aug 2009.

[69] J. Cavers. Variable-rate transmission for rayleigh fading channels. *IEEE Transactions on Communications*, 20(1):15–22, Feb 1972.

[70] B. Vucetic. An adaptive coding scheme for time-varying channels. *IEEE Transactions on Communications*, 39(5):653–663, May 1991.

[71] Tim R. Newman, Brett A. Barker, Alexander M. Wyglinski, Arvin Agah, Joseph B. Evans, and Gary J. Minden. Cognitive engine implementation for wireless multicarrier transceivers. *Wireless Communications and Mobile Computing*, 7(9):1129–1142, 2007.

[72] Magnus Nrgaard, O. E. Ravn, N. K. Poulsen, and L. K. Hansen. *Neural Networks for Modelling and Control of Dynamic Systems: A Practitioner's Handbook*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1st edition, 2000.

[73] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[74] Wei Lyu, Zhaoyang Zhang, Chunxu Jiao, Kangjian Qin, and Huazi Zhang. Performance evaluation of channel decoding with deep neural networks. *arXiv preprint arXiv:1711.00727*, 2017.

[75] Shiva Navabi, Chenwei Wang, Ozgun Y Bursalioglu, and Haralabos Papadopoulos. Predicting wireless channel features using neural networks. *arXiv preprint arXiv:1802.00107*, 2018.

[76] Tie Luo and Sai G. Nagarajan. Distributed anomaly detection using autoencoder neural networks in wsn for iot. *IEEE International Conference on Communications*, 2018.

[77] V. K. Tumuluru, P. Wang, and D. Niyato. A neural network based spectrum prediction scheme for cognitive radio. In *2010 IEEE International Conference on Communications*, pages 1–5, May 2010.

[78] Yiding Yu, Taotao Wang, and Soung Chang Liew. Deep-reinforcement learning multiple access for heterogeneous wireless networks. *IEEE International Conference on Communications*, 2018.

[79] Antti Sorjamaa, Jin Hao, Nima Reyhani, Yongnan Ji, and Amaury Lendasse. Methodology for long-term prediction of time series. *Neurocomputing*, 70(16-18):2861–2869, 2007.

[80] Simon Haykin and Xiao Bo Li. Detection of signals in chaos. *Proceedings of the IEEE*, 83(1):95–122, 1995.

[81] Jose C Principe, Neil R Euliano, and W Curt Lefebvre. *Neural and adaptive systems: fundamentals through simulations*, volume 672. Wiley New York, 2000.

[82] Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2): 179–211, 1990.

[83] Barak A Pearlmutter. Gradient calculations for dynamic recurrent neural networks: A survey. *IEEE Transactions on Neural networks*, 6(5):1212–1228, 1995.

[84] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[85] Tsungnan Lin, Bill G Horne, and C Lee Giles. How embedded memory in recurrent neural network architectures helps learning long-term temporal dependencies. *Neural Networks*, 11(5):861–868, 1998.

[86] Tsungnan Lin, Bill G Horne, Peter Tino, and C Lee Giles. Learning long-term dependencies in narx recurrent neural networks. *IEEE Transactions on Neural Networks*, 7(6):1329–1338, 1996.

[87] Eugen Diaconescu. The use of narx neural networks to predict chaotic time series. *WSEAS Transactions on Computer Research*, 3(3):182–191, 2008.

[88] Jose Maria P. Menezes and Guilherme A. Barreto. Long-term time series prediction with the narx network: An empirical evaluation. *Neurocomputing*, 71(16):3335 – 3343, 2008. Advances in Neural Information Processing (ICONIP 2006) / Brazilian Symposium on Neural Networks (SBRN 2006).

[89] Héctor Allende, Claudio Moraga, and Rodrigo Salas. Artificial neural networks in time series forecasting: A comparative analysis. *Kybernetika*, 38(6):685–707, 2002.

[90] Tsung-Nan Lin, C Lee Giles, Bill G Horne, and Sun-Yuan Kung. A delay damage model selection algorithm for narx neural networks. *IEEE Transactions on Signal Processing*, 45(11):2719–2730, 1997.

[91] I. J. Leontarits and S. A. Billings. Input-output parametric models for non-linear systems part i: deterministic non-linear systems. *International Journal of Control*, 41(2):303–328, 1985.

[92] Hossein Mirzaee. Long-term prediction of chaotic time series with multi-step prediction horizons by a neural network with levenberg–marquardt learning algorithm. *Chaos, Solitons & Fractals*, 41(4): 1975–1979, 2009.

[93] Eugen Diaconescu. The use of narx neural networks to predict chaotic time series. *Wseas Transactions on computer research*, 3(3):182–191, 2008.

[94] 3GPP TSG-RAN#148. LTE physical layer framework for performance verification, Document R1-070674. Technical report, 2007.

[95] G. J. Foschini and Z. Miljanic. A simple distributed autonomous power control algorithm and its convergence. *IEEE Transactions on Vehicular Technology*, 42(4):641–646, Nov 1993.

[96] Robert C Streijl, Stefan Winkler, and David S Hands. Mean opinion score (mos) revisited: methods and applications, limitations and alternatives. *Multimedia Systems*, 22(2):213–227, 2016.

[97] John G Proakis and Masoud Salehi. *Digital communications*, volume 4. McGraw-hill New York, 2001.

[98] Manal El Tanab and Walaa Hamouda. Resource allocation for underlay cognitive radio networks: A survey. *IEEE Communications Surveys & Tutorials*, 19(2):1249–1276, 2016.

[99] Bo Guan and Yifeng He. Optimal resource allocation for video streaming over cognitive radio networks. In *2011 IEEE 13th International Workshop on Multimedia Signal Processing*, pages 1–6. IEEE, 2011.

[100] Zhangyu Guan, Lei Ding, Tommaso Melodia, and Dongfeng Yuan. On the effect of cooperative relaying on the performance of video streaming applications in cognitive radio networks. In *2011 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2011.

[101] Sudhir Singh, Paul D Teal, Pawel A Dmochowski, and Alan J Coulson. Interference management in cognitive radio systems with feasibility detection. *IEEE transactions on vehicular technology*, 62(8): 3711–3720, 2013.

[102] Eleftherios Karipidis, Erik G Larsson, and Kaj Holmberg. Optimal scheduling and qos power control for cognitive underlay networks. In *2009 3rd IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pages 408–411. IEEE, 2009.

[103] Yonghong Zhang and Cyril Leung. Cross-layer resource allocation for mixed services in multiuser ofdm-based cognitive radio systems. *IEEE Transactions on Vehicular Technology*, 58(8):4605–4619, 2009.

[104] Zan Yang and Xiaodong Wang. Scalable video broadcast over downlink mimo–ofdm systems. *IEEE transactions on circuits and systems for video technology*, 23(2):212–223, 2012.

[105] Mohammud Z Bocus, Justin P Coon, C Nishan Canagarajah, Joe P McGeehan, Simon MD Armour, and Angela Doufexi. Resource allocation for ofdma-based cognitive radio networks with application to h. 264 scalable video transmission. *EURASIP Journal on wireless communications and networking*, 2011(1):245673, 2011.

[106] Hadi Saki and Mohammad Shikh-Bahaei. Cross-layer resource allocation for video streaming over ofdma cognitive radio networks. *IEEE Transactions on Multimedia*, 17(3):333–345, 2015.

[107] Gary Boudreau, John Panicker, Ning Guo, Rui Chang, Neng Wang, and Sophie Vrzic. Interference coordination and cancellation for 4g networks. *IEEE Communications Magazine*, 47(4):74–81, 2009.

[108] Karama Hamdi, Wei Zhang, and Khaled Ben Letaief. Joint beamforming and scheduling in cognitive radio networks. In *IEEE GLOBECOM 2007-IEEE Global Telecommunications Conference*, pages 2977–2981. IEEE, 2007.

[109] Konstantinos Ntougias, Dimitrios K Ntaikos, Constantinos B Papadias, and Georgios K Papageorgiou. Coordinated hybrid precoding and qos-aware power allocation for underlay spectrum sharing with load-controlled antenna arrays. In *2019 IEEE 20th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pages 1–5. IEEE, 2019.

[110] Suren Dadallage, Changyan Yi, and Jun Cai. Joint beamforming, power, and channel allocation in multiuser and multichannel underlay miso cognitive radio networks. *IEEE Transactions on Vehicular Technology*, 65(5):3349–3359, 2015.

[111] Manal El Tanab, Yasmine Fahmy, and Mohamed M Khairy. Opportunistic splitting algorithm for underlay cognitive radio networks. In *2014 Sixth International Conference on Ubiquitous and Future Networks (ICUFN)*, pages 9–14. IEEE, 2014.

[112] Hano Wang, Jemin Lee, Sungtae Kim, and Daesik Hong. Capacity of secondary users exploiting multispectrum and multiuser diversity in spectrum-sharing environments. *IEEE Transactions on Vehicular Technology*, 59(2):1030–1036, 2009.

[113] Shoaib Khan, Svetoslav Duhovnikov, Eckehard Steinbach, and Wolfgang Kellerer. Mos-based multiuser multiapplication cross-layer optimization for mobile multimedia communication. *Advances in Multimedia*, 2007, 2007.

[114] Tigang Jiang, Honggang Wang, and Athanasios V Vasilakos. Qoe-driven channel allocation schemes for multimedia transmission of priority-based secondary users over cognitive radio networks. *IEEE*

Journal on Selected Areas in Communications, 30(7):1215–1224, 2012.

[115] Pejman Goudarzi. A fuzzy admission control scheme for high quality video delivery over underlay cognitive radio. *Physical Communication*, 7:134–144, 2013.

[116] Bin Liu and Lei He. Qoe-based resource allocation for mixed services over cognitive radio networks. In *2016 International Conference on Computing, Networking and Communications (ICNC)*, pages 1–5. IEEE, 2016.

[117] Xiaoli He, Hong Jiang, Yu Song, Xiufeng Yang, and He Xiao. Optimal resource allocation for energy harvesting cognitive radio network with q learning. In *International Conference on Artificial Intelligence and Security*, pages 548–560. Springer, 2019.

[118] Andres Kwasinski and Wenbo Wang. Cooperative learning for reduced complexity cross-layer cognitive radio. In *2011 IEEE 22nd International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 374–378. IEEE, 2011.

[119] Weijun Zheng, Guoqing Wu, Wenbo Qie, and Yong Zhang. Deep reinforcement learning for joint channel selection and power allocation in cognitive internet of things. In *International Conference on Human Centered Computing*, pages 683–692. Springer, 2019.

[120] Snehal Sudhir Chitnavis. Cross layer routing in cognitive radio network using deep reinforcement learning. 2018.

[121] Yihang Du, Fan Zhang, and Lei Xue. A kind of joint routing and resource allocation scheme based on prioritized memories-deep q network for cognitive radio ad hoc networks. *Sensors*, 18(7):2119, 2018.

[122] Ognjen Dobrijevic, Andreas J Kassler, Lea Skorin-Kapov, and Maja Matijasevic. Q-point: Qoe-driven path optimization model for multimedia services. In *International Conference on Wired/Wireless Internet Communications*, pages 134–147. Springer, 2014.

[123] Hamid R Sheikh, Muhammad F Sabir, and Alan C Bovik. A statistical evaluation of recent full reference image quality assessment

algorithms. *IEEE Transactions on image processing*, 15(11):3440–3451, 2006.

[124] RECOMMENDATION ITU-R BT. Methodology for the subjective assessment of the quality of television pictures. *International Telecommunication Union*, 2002.

[125] Philippe Hanhart and Touradj Ebrahimi. Calculation of average coding efficiency based on subjective quality scores. *Journal of Visual Communication and Image Representation*, 25(3):555–564, 2014.

[126] B. Mobile. Mobile analytica report.

[127] Fatemeh Shah Mohammadi and Andres Kwasinski. Neural network cognitive engine for autonomous and distributed underlay dynamic spectrum access. *arXiv preprint arXiv:1806.11038*, 2018.

[128] Rui Zhang. On active learning and supervised transmission of spectrum sharing based cognitive radios by exploiting hidden primary radio feedback. *IEEE Trans. on Comm.*, 58(10):2960 –2970, Oct. 2010.

[129] Xiaoxin Qiu and Kapil Chawla. On the performance of adaptive modulation in cellular systems. *IEEE transactions on Communications*, 47(6):884–895, 1999.

[130] Quan Huynh-Thu and Mohammed Ghanbari. The accuracy of psnr in predicting video quality for different video scenes and frame rates. *Telecommunication Systems*, 49(1):35–48, 2012.

[131] Pasquale Fiengo, Giovanni Giambene, and Edmondo Trentin. Neural-based downlink scheduling algorithm for broadband wireless networks. *Computer communications*, 30(2):207–218, 2007.

[132] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[133] Christian Mehlführer, Martin Wrulich, Josep Colom Ikuno, Dagmar Bosanska, and Markus Rupp. Simulating the long term evolution physical layer. In *2009 17th European Signal Processing Conference*, pages 1471–1478. IEEE, 2009.